AFRL-RY-WP-TR-2010-1209

ADVANCED TECHNIQUES FOR SCENE ANALYSIS

Dapeng Wu

University of Florida

JUNE 2010
Final Report

AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YY)* June 2010	**2. REPORT TYPE** Final	**3. DATES COVERED *(From - To)*** 17 June 2006 – 12 June 2010

4. TITLE AND SUBTITLE ADVANCED TECHNIQUES FOR SCENE ANALYSIS	**5a. CONTRACT NUMBER** FA8650-06-1-1027
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 62204F
6. AUTHOR(S) Dapeng Wu	**5d. PROJECT NUMBER** 6095
	5e. TASK NUMBER 04
	5f. WORK UNIT NUMBER 6095041E
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Florida, Department of Electrical & Computer Engineering, 431 Engineering Building, P.O. Box 116130, Gainesville, FL 32611-6130	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory, Sensors Directorate, Wright-Patterson Air Force Base, OH 45433-7320, Air Force Materiel Command, United States Air Force	**10. SPONSORING/MONITORING AGENCY ACRONYM(S)** AFRL/RYAT
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S) AFRL-RY-WP-TR-2010-1209

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES
PAO Case Number: 88ABW 10-4190; Clearance Date: 10 Aug 2010. This report contains color.

14. ABSTRACT
The overall objective of this project is to develop advanced techniques for scene analysis. Specifically, the problems of image registration, tracking, and change detection are considered. In this project, the following new techniques have been developed: 1) robust feature-based algorithm for object tracking, 2) motion-segmentation-based technique for change detection, 3) a target detection algorithm that consists of image differencing, maximum-margin classifier, and diversity combining, 4) a rotation-invariant transform for change detection, 5) a depth-based image registration algorithm, 6) an image registration algorithm that leverages wavelet, 7) a machine learning algorithm to automatically recover 3D surface from sparse 3D points, 8) an automatic surface fitting method for 3D reconstruction from 2D video sequence, and 9) a depth-based image registration method via geometric segmentation.

15. SUBJECT TERMS
image registration, object tracking, change detection

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	SAR	190	Olga Mendoza-Schrock **19b. TELEPHONE NUMBER** *(Include Area Code)* N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

Contents

List of Figures

List of Tables

1 Summary

The overall objective of this project is to develop advanced techniques for scene analysis. For a set of images of the same scene taken at several different times, our goal is to identify the set of pixels that are ”significantly different” between the current image and the previous images; these pixels comprise the change mask. The change mask may result from a combination of underlying factors, including appearance or disappearance of objects, motion of objects relative to the background, or shape changes of objects. Specifically, we consider the problems of image registration, tracking, and change detection.

In this project, we have developed the following new techniques:

- robust feature-based algorithm for object tracking,
- motion-segmentation-based technique for change detection,
- a target detection algorithm that consists of image differencing, maximum-margin classifier, and diversity combining,
- a rotation-invariant transform for change detection,
- a depth-based image registration algorithm,
- an image registration algorithm that leverages wavelet,
- a machine learning algorithm to automatically recover 3D surface from sparse 3D points,
- an automatic surface fitting method for 3D reconstruction from 2D video sequence,
- a depth-based image registration method via geometric segmentation.

2 Introduction

To achieve the research objective, we have conducted ten research tasks. Yearly research tasks are listed as follows:

Tasks in Year 1:

Task 1 Devise novel feature-based algorithms, implement the feature-based algorithms and evaluate their performance using data in the public domain.

Task 2 Address the image registration issue in change detection.

Tasks in Year 2:

Task 3 Devise feature-based algorithms to track vehicles in urban environments. The tracking algorithm is intended to address the following challenges:

- Occlusion or partial occlusion
- Re-appearance
- A vehicle makes a left/right turn
- A vehicle passes another vehicle in the same direction
- Tracking of a small imaged object (consisting of 4 to 10 pixels) or an object of low resolution
- Image intensity change due to shadow caused by high-rise buildings
- Image intensity change due to cloud effect
- Tracking of many vehicles (e.g., thousands of vehicles in a city)
- Clutters
- Camera motion

Task 4 Design supervised learning algorithms for change detection and test our algorithms using the VHF change detection problem set, which is described at https://www.sdms.afrl.af.mil/datasets/vhf_change_detection/index.php.

Task 5 Design rotation-invariant transform for change detection and test our algorithms using the VHF change detection problem set, which is described at https://www.sdms.afrl.af.mil/datasets/vhf_change_detection/index.php.

Tasks in Year 3:

Task 6 Devise an image registration algorithm that is capable of mitigating the parallax problem.

Task 7 Develop a wavelet based image registration scheme that can achieve low Root Mean Squared Error (RMSE) in registration.

Tasks in Year 4:

Task 8 Develop a machine learning algorithm to automatically recover 3D surface from sparse 3D points.

Task 9 Design an automatic surface fitting method for 3D reconstruction from 2D video sequence.

Task 10 Develop a depth-based image registration method via geometric segmentation.

Next we present what we have accomplished for each of the ten tasks from Section 3 through Section 12.

3 Robust Feature-based Object Tracking

The goal of Task 1 is to devise novel feature-based algorithms, implement the feature-based algorithms and evaluate their performance using data in the public domain. For Task 1, we address the object tracking problem since object tracking is an important component of many computer vision systems. It is widely used in video surveillance, robotics, 3D image reconstruction, medical imaging, and human computer interface. In this task, we focus on unsupervised object tracking, i.e., without prior knowledge about the object to be tracked. To address this problem, we take a feature-based approach, i.e., using feature points (or landmark points) to represent objects. Feature-based object tracking consists of feature extraction and feature correspondence. Feature correspondence is particularly challenging since a feature point in one image may have many similar points in another image, resulting in ambiguity in feature correspondence. To resolve the ambiguity, algorithms, which use exhaustive search and correlation over a large neighborhood, have been proposed. However, these algorithms incur high computational complexity, which is not suitable for real-time tracking. In contrast, Tomasi and Kanade's tracking algorithm only searches corresponding points in a small candidate set, which significantly reduces computational complexity; but the algorithm may lose track of feature points in a long image sequence. To mitigate the limitations of the aforementioned algorithms, we propose an efficient and robust feature-based tracking algorithm. The key idea of our algorithm is as below. For a given target feature point in one frame, we first find a corresponding point in the next frame, which minimizes the sum-of-squared-difference (SSD) between the two points; then we test whether the corresponding point gives large value under the so-called Harris criterion. If not, we further identify a candidate set of feature points in a small neighborhood of the target point; then find a corresponding point from the candidate set, which minimizes the SSD between the two points. The algorithm may output no corresponding point due to disappearance of the target point. Our algorithm is capable of tracking feature points and detecting occlusions/uncovered regions. Experimental results demonstrate the superior performance of the proposed algorithm over the existing methods.

The results of Task 1 were published in Ref. [1]. Next, we present the technical details.

3.1 Introduction

Accurate tracking of feature points over image sequences is a critical and essential process for vision systems ranging from unmanned aerial vehicles to medical devices. Study has shown that establishing correspondence between image patches, through correlation-based measures or sum-of-squared differences (SSD), can achieve effective feature tracking.

A feature-based tracking algorithm must first assume a form to model an object's motion. Traditionally, motion has been represented as translational, which indeed proves reliable for small, linear movements. When tracking over a longer image sequence, however, more complex models are needed as geometric deformations of objects become significant. Shi and Tomasi noted in [2] that translational models are poor representations when an object has undergone an affine transformation. In their approach, translational models are used for tracking, which provides higher reliability and accuracy over smaller inter-frame camera motions. To monitor the image sequence for occlusions via comparisons between the first and current frames, affine models are utilized, which serve to better account for longer object deformations that potentially could have occurred.

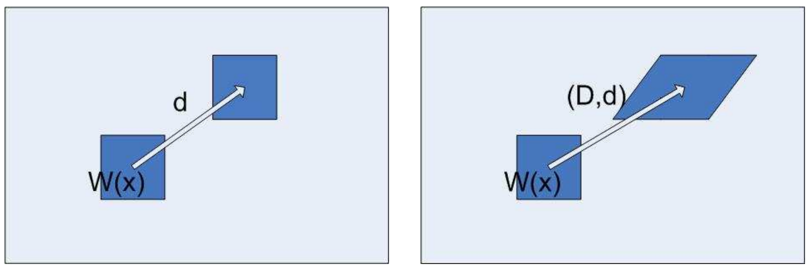
Once a motion model has been established, an object can be tracked over frames through SSD or correlation methods. To ensure that the accuracy of tracked features is maintained and to reject outliers that arise through occlusion, a successful tracking algorithm must adopt a criterion through which to monitor a feature's quality. In [2], Shi and Tomasi used a measurement of features' rms residue between the initial and current frame, which they described as "dissimilarity", to quantify the change in a feature's appearance over frames. Jin, Favaro, and Soatto [3] proposed a rejection rule based on a normalized cross-correlation function which furthermore compensated for differences in intensity variation among the patches of interest. In order to achieve stability and robustness against occlusions, several other tracking methods [4], [5] have used Kalman filters to smooth the object trajectory and monitor the object's motion.

Feature correspondence presents a challenging problem for feature-based object tracking, as ambiguity often arises when a feature point in one image has many similar points in another image. To alleviate ambiguity, algorithms often perform an exhaustive search or compute pixel correlations over large windows. As a result, the computational complexity of the algorithms is considerably increased. In contrast, Tomasi and Kanade in [6] use small windows to track the translational motion of objects by minimizing a residue error, thus reducing complexity. However, their approach may lose a significant percentage of tracked points over longer sequences.

To prevent this loss of feature points, our algorithm couples the approach taken by Tomasi and Kanade with an SSD criterion. Furthermore, to ensure robustness but still avoid computational complexity, our method employs the combined motion model described in [2].

Depending on the application, not all feature points in an image sequence necessarily provide "useful" information to the researcher. To obtain segmentation of objects and the image background, our algorithm uses the greedy learning procedure developed by Williams and Titsias in [7]. Their learning model sequentially extracts object parameters from a sequence using a robust statistical approach, and thus avoids the complexity that often results from segmentation algorithms. The approach taken by Williams and Titsias stems from an earlier model described by Jojic and Frey in [8], which conversely learns objects simultaneously using a variational method. By first segmenting the objects from the sequence, our algorithm is able to filter out undesirable features and thus ensure that all of the tracked points are indeed meaningful.

The remainder of this section is organized as follows. In Section 3.2, we introduce the translational and affine image motion models, and Shi-Tomasi's [2] combined motion model. Section 3.3 explains the object segmentation method used in our algorithm. In Section 3.4, we present our point feature tracking algorithm. Section 3.5 shows the experimental results and a performance evaluation of the proposed algorithm versus previous approaches.



(a) (b)

Figure 1: (a) Translational motion deformation of local domain **W(x)**, and (b) affine motion deformation of local domain **W(x)**.

3.2 Image Motion Models

In order to track a point feature in a video sequence, an image motion model should first be defined. The image motion model relates information about the image's deformation. Generally, either a translational model or an affine motion model is used.

3.2.1 Translational motion model

Image intensity patterns change in an intricate way. In the translational motion model, however, we only concentrate on the "regions" of the image that can be modeled as undergoing a linear transformation. Translational motion models assume that each point in the window undergoes identical motion. Translational models for point feature tracking are easy to implement and computationally costless. Fig. 1(a) shows the translational motion of a fixed window **W(x)**.

Let **x** be the central point of the concentrated "region". Let **W(x)** be the window around **x**. Let the function **h** describe the transformation of the image motion. We have

$$\mathbf{h}(\tilde{\mathbf{x}}) = \tilde{\mathbf{x}} + \Delta\mathbf{x}, \tag{1}$$

where, $\forall\tilde{\mathbf{x}} \in \mathbf{W}(\mathbf{x})$, $\Delta\mathbf{x} \in \mathbb{R}^2$. This model is valid for portions of a scene that are flat and parallel, and whose movements are parallel, to the image plane. The approximation applies only locally in space and in time. Although coarse, the model is at the core of most feature matching or tracking algorithms due to its simplicity and the efficiency of its implementation.

3.2.2 Affine motion model

When variations of the displacement vector are noticeable even within the small windows used for tracking, translational models fail to describe the event, as different motions occur within the same window. An affine motion model, shown in Fig. 1(b), is thus introduced.

More precisely, an affine motion model is defined as:

$$\mathbf{h}(\tilde{\mathbf{x}}) = \mathbf{D}\tilde{\mathbf{x}} + \mathbf{d}, \tag{2}$$

where $\mathbf{D} \in \mathbb{R}^{2\times 2}$ and $\mathbf{d} \in \mathbb{R}^2$. $\mathbf{D}$ is a deformation matrix and $\mathbf{d}$ is the translation of the center of the window. The quality of this estimation depends on the size of the feature window, the texture of the image within it, and the amount of camera motion between frames. This model serves as a good approximation for small planar patches parallel to the image plane that undergo an arbitrary translation and rotation about the optical axis, and only a modest rotation about an axis orthogonal to the plane. The affine model represents a convenient tradeoff between simplicity and flexibility. In the prevailing algorithms, an affine motion model is used to identify the good features.

3.2.3 Combined motion model

Both translational models and affine models have limitations, and thus a combination of the two models will be established. In Shi and Tomasi's paper [2] , they monitor the quality of image features during tracking by measuring features' rms residues between the first and the current frame. When the dissimilarity becomes too large, the feature will be abandoned. For this case, affine models, rather than translational, prove more suitable. When the window is small, the matrix **D** is harder to estimate, as the variations of motion within the window are smaller and therefore less reliable.

Generally, smaller windows are preferable for tracking because they are less likely to straddle a depth discontinuity. Whereas an affine motion is used for comparing features between the first and the current frame, a translational model is preferable during tracking for its improved accuracy and reliability.

3.3 Object Segmentation

In this section, we will briefly describe the approach to object segmentation taken by Titsias and Williams in [7]. For our algorithm, object segmentation will be used to isolate features of interest for tracking.

3.3.1 Segmentation model

The goal of the algorithm is to learn appearance-based models to describe the background and foreground objects in an image. The variable **f** will be used to describe the appearance of the foreground object, where, for brevity, we will assume in this explanation that only one object is present. A vector of probabilities, $\boldsymbol{\pi}$, will be assigned to the image, where $\pi_j \approx 1$ when pixel j is part of the foreground object and $\pi_j \approx 0$ otherwise. The background appearance will be described by **b**. A single frame can then be described by the following mixture distribution:

$$p(\mathbf{x}) = \prod_{p=1}^{P} [\pi_p p_f(x_p; f_p) + (1 - \pi_p) p_b(x_p; b_p)], \tag{3}$$

where P denotes the number of pixels in the image, $p_f(x_p; f_p) = N(x_p; f_p; \sigma_f^2)$, $p_b(x_p; b_p) = N(x_p; b_p; \sigma_b^2)$, and $N(x; \mu; \sigma^2)$ represents a Gaussian distribution with mean μ and variance σ^2.

To account for the object motion over frames, we let j_f and j_b denote the object and background transformation, respectively. If, assuming only translations, we then let the matrix $\mathbf{T}_{j_f}$ represent the transformation j_f and $\mathbf{T}_{j_b}$ represent j_b, then:

$$\begin{aligned} p(\mathbf{x}|j_f, j_b) \quad = \quad & \prod_{p=1}^{P} [(T_{j_f}\boldsymbol{\pi})_p p_f(x_p; (T_{j_f}\mathbf{f})_p) + \\ & (\mathbf{1} - T_{j_f}\boldsymbol{\pi})_p p_b(x_p; (T_{j_b}\mathbf{b})_p)]. \end{aligned} \tag{4}$$

The parameters needed to model the scene, given by $\theta = \{\mathbf{f},\ \boldsymbol{\pi}, \mathbf{b}, \sigma_f^2, \sigma_b^2\}$, can be obtained by maximizing the likelihood $L(\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}^n|\theta)$ using the EM algorithm, where j_f and j_b are the unknown parameters. To reduce complexity, Williams and Titsias extract the background and foreground appearances sequentially.

3.3.2 Learning the background

To obtain the background of the sequence, the foreground mask $\boldsymbol{\pi}$ is effectively set to zero so that (4) becomes:

$$p(\mathbf{x}) = \prod_{p=1}^{P} [p_b(x_p; b_p)]. \tag{5}$$

The log likelihood of the background is then:

$$L_b = \sum_{n=1}^{N} \log \sum_{j_b=1}^{J} P_{j_b} p(\mathbf{x}^n | j_b), \tag{6}$$

which is then maximized using the EM algorithm to obtain j_b.

3.3.3 Learning the foreground

After finding the background, the foreground mask $\boldsymbol{\pi}$ in (4) is then allowed to take on non-zero values. As a direct maximization over the new likelihood would require a search over $J_f \times J_b$ possibilities, Williams and Titsias instead use the constrained EM algorithm presented in [9]; the computational complexity is reduced to J_f by using the background transformation already obtained. By introducing a distribution $Q^n(j_f, j_b)$ and using Jensen's inequality, they produce a lower bound on the likelihood:

$$\begin{aligned} F &= \sum_{n=1}^{N} \sum_{j_b j_f} Q^n(j_f|j_b) Q^n(j_b) \{\log[P_{j_f} P_{j_b} \\ &\times \prod_{p=1}^{P} p(x_p^n | j_b, j_f)] - \log[Q^n(j_f|j_b) Q^n(j_b)]\}, \end{aligned} \tag{7}$$

which can be tightened by setting $Q^n(j_b, j_f) = P(j_b, j_f|\mathbf{x}^n)$ for every image $\mathbf{x}^n$. Maximization can then be performed, where in the expectation step F is maximized with respect to the Q^n distribution and in the maximization step F is maximized with respect to the object parameters $\{\mathbf{f}$, $\boldsymbol{\pi}$, $\sigma_f^2\}$. The update equations are omitted here, but can be found in [7].

3.4 Tracking Algorithms

Robustness and accuracy are two important criterions for a tracking algorithm. In addition, feature tracking demands a computationally efficient approach. In this section, we will first describe several basic feature-based algorithms. Then, we will present our new algorithm, which will seek to achieve better efficiency and performance than previous approaches.

3.4.1 Feature selection

In the first step of feature selection, candidate features in one or more frames from the given sequence of images are selected. No feature-based algorithm can work unless good features can be identified first. For the case of point features, a popular algorithm is the well-known Harris corner detector. The quality of a point with coordinates $\mathbf{x} = [x, y]^T$, as a candidate feature, can be measured by

$$C(\mathbf{x}) = \det(\mathbf{G}) + k \times \text{trace}^2(\mathbf{G}), \tag{8}$$

computed on the window $\mathbf{W}(\mathbf{x})$. In the equation, $\mathbf{G}$ is a 2×2 matrix which depends on $\mathbf{x}$, given by

$$\mathbf{G} = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}, \tag{9}$$

where I_x and I_y are the gradients obtained by convolving the image I with the derivatives of a pair of Gaussian filters and k is a constant parameter that can be chosen by the user.

A point feature is selected if $\mathrm{C}(\mathbf{x})$ exceeds a certain threshold τ. In this way, a feature is selected only if the window contains "sufficient texture". In order to achieve tracking efficiency, we do not want the features to be concentrated in a small region within the whole image. However, in some specific feature-rich regions, more than one feature may be selected out, which does not prove efficient for tracking. To mediate this problem, we define a minimal space **p** between two selected features, such that a candidate feature point should be sufficiently distanced from other selected points.

Fig. 2 and Fig. 3 show two sets of initial features selected from the same image. A small minimal space **p** results in more features which serve to describe the same region. Generally, we choose **p** to be 5 or 6 in order to achieve a better tradeoff between robustness and efficiency.

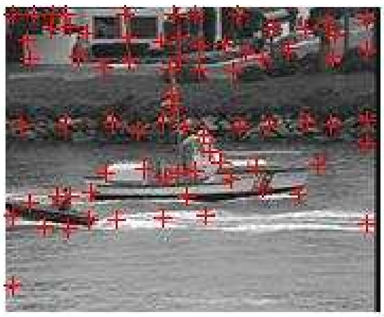


Figure 2: An example of the response of the Harris feature detector using **p** = 6.

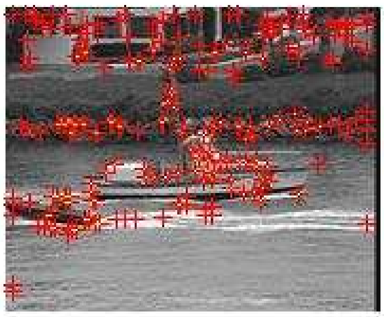


Figure 3: An example of the response of the Harris feature detector using **p** = 3.

3.4.2 Sum-of-squared-difference criterion

Under the assumption of the simple translational deformation model, tracking a point feature **x** is the process of finding out the location $\mathbf{x} + \triangle\mathbf{x}$ on the frame at time $t + \tau$ whose window is most similar to the window $\mathbf{W}(\mathbf{x})$.

A common way of measuring similarity is by using the sum-of-squared-difference(SSD) criterion. The SSD criterion compares the image window **W** centered at the location (*x*, *y*) at time *t* and other candidate locations (*x* + *dx*, *y* + *dy*) on the frame at time *t* + *dt*, where the point could have moved between the two frames. The displacement **d** is obtained by minimizing the SSD criterion

$$E_t(dx, dy) \doteq \sum [I(x + dx, y + dy, t + dt) - I(x, y, t)]^2, \tag{10}$$

where the subscript t indicates the translational deformation model.

One alternative way to compute the displacement **d** is to evaluate the function at each location and choose the one that gives the minimum error. This formulation is due to Lucas and Kanade [10], and was originally proposed in the context of stereo algorithms. The algorithm was later refined by Tomasi and Kanade [6] in a more general feature-tracking context.

3.4.3 Pyramidal decomposition

Multi-scale decomposition is a widespread tool in image processing. The typical recursive form of the algorithm, which decomposes signals into information at different levels, leads to large improvements in computational efficiency. Simoncelli and Freeman [11] proposed a steerable pyramid for efficient and accurate linear decomposition of an image into scale and orientation. Bouguet developed an algorithm to implement pyramidal image scales of the Lucas-Kanade feature tracker.

The proposed tracking scheme is implemented in a multi-scale fashion by constructing a pyramid of images through smoothing and downsampling of the original image. For instance, let I^0 be the original image and I^{L-1} represent the image at level $L - 1$. The L^{th} level image is then defined as follows:

$$\begin{aligned} I^L(x,y) =& \frac{1}{4} I^{L-1}(2x,2y) + \\ & \frac{1}{8}(I^{L-1}(2x-1,2y) + I^{L-1}(2x+1,2y) \\ & + I^{L-1}(2x,2y-1) + I^{L-1}(2x,2y+1)) + \\ & \frac{1}{16}(I^{L-1}(2x-1,2y-1) + I^{L-1}(2x+1,2y+1) \\ & + I^{L-1}(2x-1,2y-1) + I^{L-1}(2x+1,2y+1)). \end{aligned} \tag{11}$$

For a given feature central point **x**, its corresponding coordinates on the pyramidal images are defined by

$$\mathbf{x}^L = \frac{\mathbf{x}}{2^L}. \tag{12}$$

Then we could compute each motion direction vector $\mathbf{d}^L$ on each level of pyramidal images. Finally, we could sum up all levels of motion vector **d** as

$$\mathbf{d} = \sum 2^L \mathbf{d}^L. \tag{13}$$

A key problem to any feature tracker is the tradeoff between accuracy and robustness. Intuitively, accuracy requires a small tracking window in order to preserve details in the image while robustness prefers a bigger intergration window to handle larger motions. The advantage of pyramidal implementation is that, while each motion vector $\mathbf{d}^L$ is obtained by way of smaller integration windows, the overall displacement vector **d** can account for larger pixel motions, thus achieving both accuracy and robustness.

Fig. 4 shows a 3-level steerable pyramid decomposition of a disk image, with $k = 1$, where k represents the number of orientation bands. Fig. 5 shows a 3-level steerable pyramid decomposition with $k = 3$. The filters are directional second derivatives oriented at $\theta \in \{-2\pi/3, 0, 2\pi/3\}$.

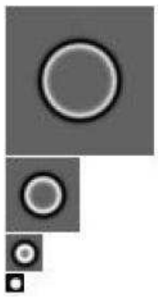


Figure 4: A 3-level, $k = 1$ steerable pyramid(non-oriented). Shown are the three bandpass images at each scale and the final lowpass image [11].

3.4.4 Improved algorithm

From our experimental data, we find out that both minimal SSD criterion or Tomasi-Kanade's algorithm [6] will lose track of more than half of the initial selected features after 15 to 30 frames. In order to increase the robustness of the tracking algorithm, we combine the two tracking methods and propose a more robust feature-based tracking algorithm.

First, we will use Harris's criterion to select the set of initial candidate features ($\mathbf{S}_1$) in the first frame of the video sequence. Each feature is then tracked using two methods of detection. Minimal SSD criterion is applied in the first step to find the most similar region of the target feature in the time adjacent frame. If the quality $C(\mathbf{x})$ of the region $\mathbf{W}(\mathbf{x} + \mathbf{d})$ exceeds the chosen threshold τ_H, it will be updated to the set of tracking features. However, when the motion becomes more complicated or after the feature is tracked through a long sequence of frames, the most similar region that fit minimal SSD criterion may not be a good feature to keep on tracking. In this case, the feature will be declared lost of tracking in the former SSD criterion algorithm. In the experiment, more than ten percent of features are lost because the quality degrades through the tracking sequence. In order to solve this problem, we will introduce Tomasi-Kanade's [6] algorithm as a complement to minimal SSD criterion.

Once the quality $C(\mathbf{x})$ of the tracked feature is lower than the threshold τ_H, we will define another set of candidate features ($\mathbf{S}_2$) to be

$$\mathbf{S}_2 = \mathbf{S}_3 - \mathbf{S}_1, \tag{14}$$

where $\mathbf{S}_3 = \{\mathbf{x}_1, ...\mathbf{x}_j, ...\mathbf{x}_m\}$ are the potential features in the tracking window $\mathbf{W}_2(\mathbf{x})$. After subtracting the points that are already in the set of initial candidate features, other features will be calculated to find the one that gives the minimal squared difference. If the similarity between the

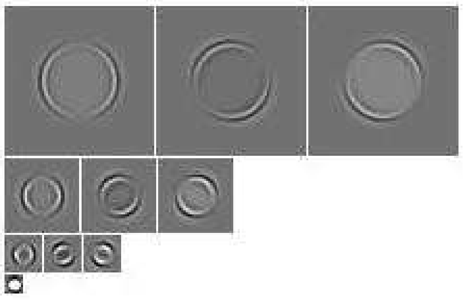


Figure 5: A 3-level, $k = 3$ steerable pyramid(second derivative). Shown are the three bandpass images at each scale and the final lowpass image [11].

two regions is smaller than the threshold τ_E, it will also be taken as a successfully tracked feature.

In this way, we can increase the robustness when tracking point features along a long image sequence. The algorithm could also perform well when the motion between two time adjacent images is not purely translational, such as through scattering or scaling.

3.5 Experiment Results

3.5.1 Algorithm

The full algorithm is presented in Fig. 6. In the code, the feature quality, $C(\mathbf{x})$, is calculated by Harris's criterion. In the event that the object's motion is known to be translational and void of scaling, the use of the purely translational model yields improved results versus the model which accounts for scaling.

3.5.2 Results

We have tested the algorithm on video clips consisting of various types of motion, including translational, affine, and scaling deformations. Features are identified in the initial frame as the candidate features to be tracked through the whole image sequence. The performance of the algorithm depends on the content of the test video.

In the experiment, parameters could be adjusted according to the type of motion in the test image sequence. For each of our simulations, the threshold for selection is $\tau_S = 0.01$, the threshold for SSD criterion in the tracking process is $\tau_E = 0.1$, and the threshold for Harris's criterion in the tracking process is $\tau_H = 10$.

```
1)  function FeatureSelection($I$)
2)      Choose window $W_1(\mathbf{x})$
3)      Compute the quality $C(\mathbf{x})$ of each pixel using eq 8
4)      Choose threshold $\tau_s$
5)      Sort $\mathbf{x}$ in decreasing order of $C(\mathbf{x})$
6)      if $C(\mathbf{x}_i) > \tau_s$
7)          if $\mathbf{x}_i - \mathbf{x}_j <$minimal space($j < i$)
8)              $S_1 \leftarrow x_i$
9)          end
10)     end
11) end function
12) function FeatureTracking($\mathbf{x}'$)
13)     Build a pyramid of k levels of images
14)     for $level = 1 : k$
15)         Compute $\mathbf{d}^k = -G^{-1}\mathbf{b}$ for the image pair $(I_1^k, I_2^k)$
16)         Move the window $W(\mathbf{x})$ by $2\mathbf{d}^k$ through warping the next level
    image $I_2^{k-1}(\mathbf{x})$
17)         Update the displacement $\mathbf{d} \leftarrow \mathbf{d} + 2\mathbf{d}^k$ and the index $k \leftarrow k$ - $1$
18)     end
19)     evaluate the quality of the feature by Harris's criterion defined by
    equation 8
20)     if $C(\mathbf{x}) > \tau_E$
21)         update $\mathbf{x}_i \leftarrow \mathbf{x}_{i+1}$
22)     else
23)         Find all the candidate features in the tracking window $W_2(\mathbf{x})$
    on image $I_2$
24)         Calculate and compare SSD for each candidate feature and the
    target feature, find the one gives minimal SSD
25)         if $MSSD < \tau_E$
26)             update $\mathbf{x}_i \leftarrow \mathbf{x}_{i+1}$
27)         else
28)             returnlost
29)         end
30)     end
31) end function
```

Figure 6: Combined feature-based tracking algorithm.

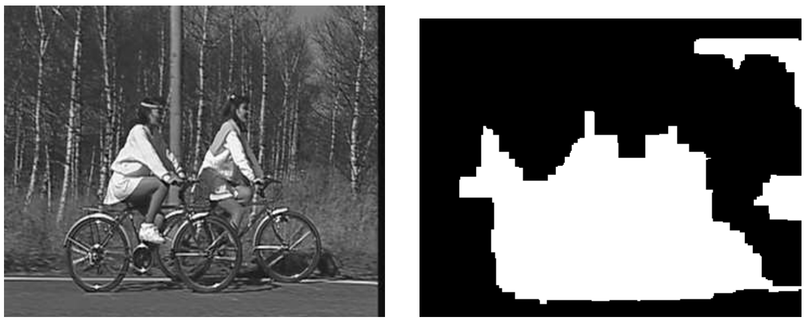


Figure 7: (a) 1st frame in the bike sequence. (b) Mask for bike sequence created using [7].

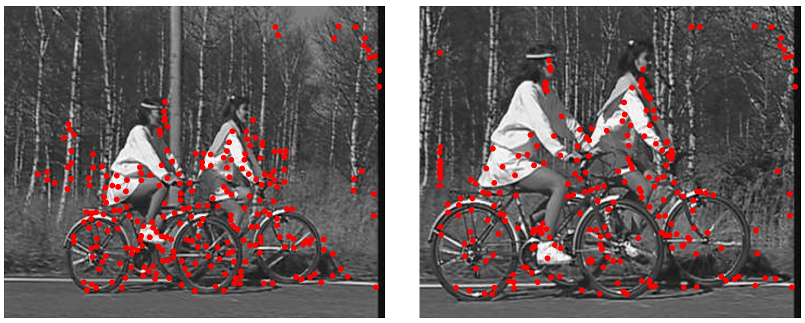


Figure 8: Image in tracked bike sequence: (a) 1^{st} frame. (b) 50^{th} frame.

To evaluate the proposed algorithm, we applied the algorithm developed by Tomasi and Kanade in [6] and the improved algorithm which combines the SSD criteria and Tomasi-Kanade's algorithm.

Fig. 7(a) shows the initial frame of a sequence displaying two bikers traveling at approximately the same speed. The sequence undergoes translational motion, affine motion, and scaling. In Fig. 7(b), the object mask created using the greedy learning approach [7] is shown, where noise in the mask has been removed by way of morphological filters. As evidenced, the mask successfully recovers both bikers, but also erroneously contains some of the background on the right side of the frame. After applying the mask to the initial frame of the sequence, the combined tracking algorithm and Tomasi-Kanade's algorithm were applied to the 50-frame video. The results from the first and last frame are shown in Fig. 8(a) and Fig. 8(b), respectively, using the combined method. Furthermore, Fig. 9 illustrates the number of features tracked during each frame of the sequence using the combined algorithm, shown with the open circles, and the Tomasi-Kanade approach, shown with the filled squares. The combined algorithm improves the number of successfully tracked features by over 10 percent.

Fig. 10(a) displays the initial frame of a sequence containing a coastguard boat on a river. The object mask created [7] is shown in Fig. 10(b). Unlike the previous video, the coastguard sequence contains only translational motion. The first frame of the tracked sequence, after applying the mask, is shown in Fig. 11 for the combined algorithm. However, as evidenced in Fig. 12, the combined algorithm proves equivalent to the results obtained using the Tomasi-Kanade approach. Thus, as expected, the combined algorithm only offers improved performance for image sequences containing affine or scaling deformations.

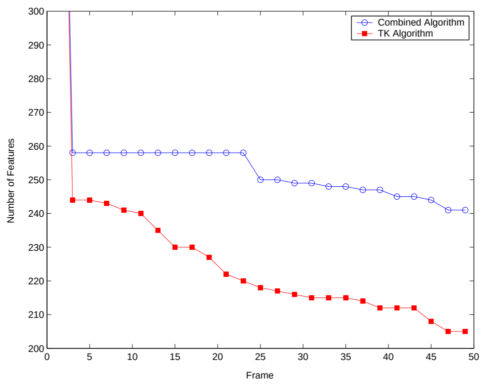


Figure 9: Number of features selected from the bike sequence using the combined algorithm and the Tomasi-Kanade(TK) algorithm.

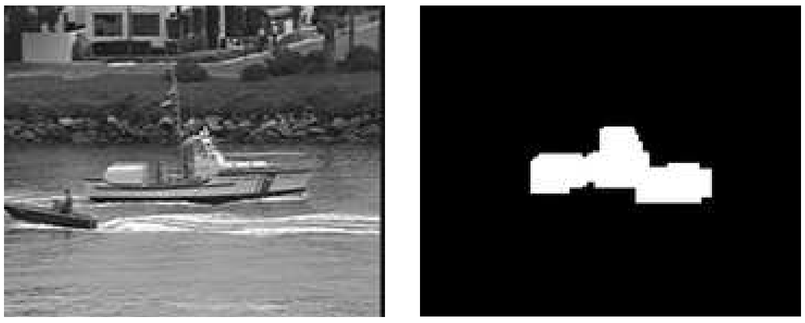


Figure 10: (a) 1st frame in the coastguard sequence. (b) Mask for coastguard sequence created using [7].

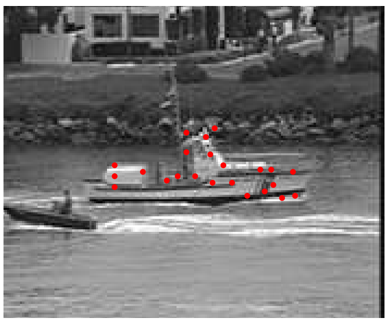


Figure 11: 1st tracked frame in the coastguard sequence using the combined algorithm.

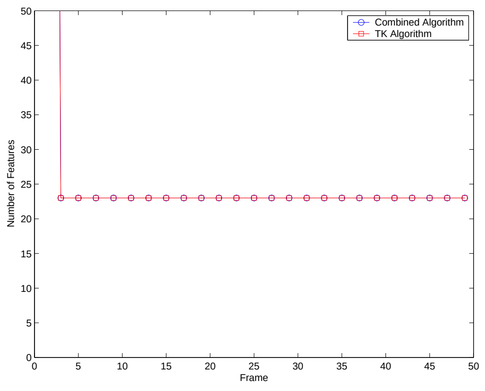


Figure 12: Number of features selected from the coastguard sequence using the combined algorithm and the Tomasi-Kanade (TK) algorithm.

4 Motion-Segmentation-Based Change Detection

The goal of Task 2 is to address the image registration issue in change detection. Detecting regions of change in images of the same scene taken at different times is of widespread interest. Important applications of change detection include video surveillance, remote sensing, medical diagnosis and treatment. Change detection usually involves image registration, which is aimed at removing meaningless changes caused by camera motion. Image registration is a hard problem due to the absence of knowledge about camera motion and objects in the scene. To address this problem, we propose a novel motion-segmentation based approach to change detection, which represents a paradigm shift. Different from the existing methods, our approach does not even need image registration since our method is able to separate global motion (camera motion) from local motion, where local motion corresponds to regions of change while regions with only global motion will be classified as ‘no change’. Hence, our approach has the advantage of robustness against camera motion.

Separating global motion from local motion is particularly challenging due to lack of prior knowledge about camera motion and the objects in the scene. To tackle this, we introduce a motion-segmentation approach based on minimization of the coding length. The key idea of our approach is as below. We first estimate the motion field by solving the optical flow equation; then we segment the motion field into regions with different motion, based on the minimum coding length criterion; after motion segmentation, we estimate the global motion and local motion; finally, our algorithm outputs regions of change, which correspond to local motion. Experimental results demonstrate the effectiveness of our scheme.

The results of Task 2 were published in Ref. [12]. Next, we present the technical details.

4.1 Introduction

Detecting regions of change in images of the same scene taken at different times is of widespread interest due to a large number of applications in diverse disciplines. Important applications of change detection include video surveillance, remote sensing, medical diagnosis and treatment, civil infrastructure, and underwater sensing. Change detection usually consists of three steps, namely, 1) image registration, a.k.a., geometric adjustment, to remove meaningless changes caused by camera motion, 2) radiometric/intensity adjustment to mitigate lighting variation and noise, and 3) stochastic modeling and hypothesis testing, to decide which pixel/area experiences changes of interest. This conventional paradigm involves image registration, which is a challenging problem to solve due to the absence of knowledge about camera motion and objects in the scene. To address this challenge, we propose a novel motion-segmentation based approach to change detection, which represents a paradigm shift. Different from the conventional paradigm, our approach dispenses with image registration since our method is able to separate global motion (camera motion) from local motion, where local motion corresponds to regions of change while regions with only global motion will be classified as 'no change'. Hence, our approach has the advantage of robustness against camera motion.

Separating global motion from local motion is particularly challenging due to lack of prior knowledge about camera motion and the objects in the scene. To tackle this, we introduce a motion-segmentation approach. Motion segmentation [13] is different image segmentation. Image segmentation methods, including edge detection and intensity thresholding, are used to provide information about a single frame [14]. For a sequence of images, a complete understanding of the scene can prove very challenging, as the occurrence of multiple moving objects or the appearance of objects at different depths could lead to motion discontinuities. To obtain needed information about the objects' movements or regions of change [15], segmentation can be used to divide a picture/frame of an image sequence into motion-based regions; this process is called motion segmentation.

Motion segmentation is a central constituent of several technologies. The MPEG-4 standard, which is used to compress digital audio and video data [16], describes a content-based manipulation of objects in image sequences. To create an object-based scene representation, different objects in a frame are segmented, typically by way of their motion information. Video querying [17], another new field, aims to automatically classify video sequences based on their content. A common video query task requires retrieving all the images in a database that have similar content to the input image. By providing an indexing scheme that uses the trajectories, shapes, and flow vectors of the moving objects, motion segmentation serves to enhance the performance of video querying. The development of unmanned aerial vehicles [18], a major advancement in reconnaissance, requires scene analysis technology to identify suspicious military vehicles in a video sequence. Objects having different moving velocities or directions can be identified and isolated through segmentation techniques.

4.1.1 Literature Review

Motion segmentation refers to the assignment of groups of pixels to various classes based on the speed and direction of their movements. Most approaches to motion segmentation first seek to compute the optical flow of the image sequence. In [19], Horn and Schunck presented an iterative method to calculate the optical flow. Using the rate of change of the image intensity and by assuming that the brightness function changes smoothly, they attained the flow velocity by minimizing a global error function. Though Horn and Schunck's algorithm achieves a dense optical flow field, their approach remains sensitive to noise. An alternate method is proposed by Lucas and Kanade in [20]. By assuming a locally constant flow, their algorithm achieves improved robustness against noise at the expense of the resolution of their optical field.

Many different methods currently exist to provide segmentation of the flow field. Wang and Adelson described in [21] a method to represent moving objects using sets of overlapping layers. The layers are obtained using a K-means clustering algorithm, and are ordered based on their depth in the image. Furthermore, a velocity map is used to describe the deformation of the layers over time. Several proposed algorithms follow this layer-based approach [22], [8]. In contrast, Schnorr and Cremers [23] presented a variational method, called motion competition, which jointly solves the problems of motion estimation and segmentation for two consecutive frames in a sequence. By their approach, a single energy functional is minimized with respect to the affine motion model in each separate region and to the shape of the separating contour.

All of the above approaches try to solve the motion segmentation problem, but only via motion information. Image intensity information could also be used to find the optimal motion segmentation results. In addition, previous methods fail to provide a robust means by which to extract the global motion from an image sequence.

4.1.2 Our Approach

In this paper, we propose an approach for motion segmentation based on an image's optical flow field. The novelty of our work involves the introduction of minimum coding length as a criterion in clustering or grouping of motion vectors so that a motion field[1] is segmented into regions of different motion. Furthermore, we propose a heuristic and highly effective method for the estimation of global motion in an image sequence. Based on our techniques for motion segmentation and global motion estimation, we design a change detection algorithm. The key idea of our change detection algorithm is as below. We first estimate the motion field by solving the optical flow equation; then we segment the motion field into regions with different motion, based on the minimum coding length criterion; after motion segmentation, we estimate the global motion and local motion; finally, our algorithm outputs regions of change, which correspond to local motion. Our experimental results demonstrate the effectiveness of our scheme.

The remainder of this section is organized as follows. In Section 4.2, we describe a method to estimate the motion field. Section 4.3 describes our motion segmentation approach, which is based on the minimum-coding-length criterion. In Section 4.4, we present our scheme for global motion estimation. Section 4.5 describes our motion-segmentation-based change detection algorithm, which is based on the schemes presented in Sections 4.2 through 4.4. Section 6.5 shows the experimental results for the proposed algorithms.

4.2 Motion Field Estimation

Optical flow is a method of estimating the motion of objects within a visual representation. Typically, the motion is represented as a vector indicating the direction and speed of a pixel as it moves across the image. It is a powerful tool for image analysis and interpretation.

The traditional approaches to computing optical flow can be classified into categories: feature-based, correlation-based, and gradient-based. Gradient-based algorithms receive special interest due to their mathematical simplicity and relative computational efficiency. In this paper, we will use a gradient-based algorithm, which was first proposed by Lucas and Kanade [20], to estimate the motion field of an image sequence.

This method seeks to calculate the motion between two image frames which are taken at times t and t + δt at every pixel position. As a pixel at location (x, y, t) with intensity $I(x, y, t)$ will have

[1]A motion field is defined as a 2D array, each element of which is a motion vector associated with an image pixel.

moved by δx, δy, and δt between the two frames, the following image constraint equation can be given:

$$I(x,y,t) = I(x+\delta x, y+\delta y, t+\delta t). \tag{15}$$

By assuming that movement of objects is small enough, the image constraint at $I(x,y,t)$ can be expanded with a Taylor series:

$$\begin{aligned} I(x+\delta x, y+\delta y, t+\delta t) &= I(x,y,t) \\ &+\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + H.O.T., \end{aligned} \tag{16}$$

where $H.O.T.$ stands for the higher ordered terms, which are ignored here. From the above equation, we can achieve

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0, \tag{17}$$

which results in

$$I_x V_x + I_y V_y = -I_t, \tag{18}$$

where V_x and V_y are the components of the velocity and I_x, I_y, and I_t are the spatial and temporal derivatives of the image at pixel (x,y,t).

This equation is known as the aperture problem of optical flow algorithms. In order to solve this problem, an additional constraint is needed. In Lucas and Kanade's solution, they use a non-iterative method that assumes a locally constant flow. From this, we are able to solve the constrained system of equations, yielding:

$$A\overrightarrow{v} = -b, \tag{19}$$

where A is the constant flow in a small window of size $m \times m$.

The above algorithm shows that optical flow can be computed by calculating the derivatives of the image in both spatial and temporal dimensions. In order to give more prominence to the center pixel in a window, a Gaussian weighting function is preferably incorporated. This flow model can be further extended to affine image deformations.

For example, Figs. 13 and 14 show two frames from an image sequence, and Fig. 15 shows the calculated optical flow field. To maintain a better visual effect, we only draw the motion vector for each 5 by 5 block. The actual motion field is calculated per pixel.

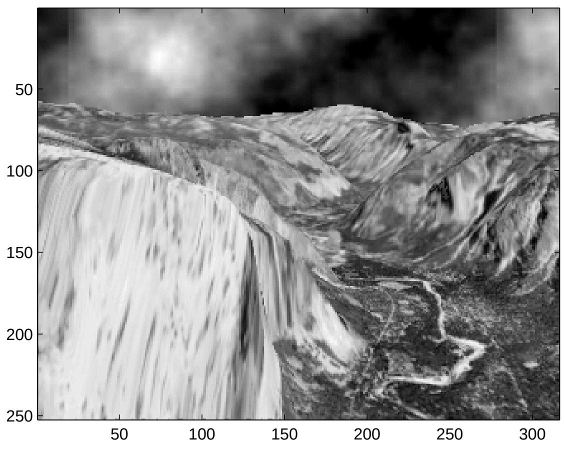


Figure 13: The 2^{nd} frame of the image sequence.

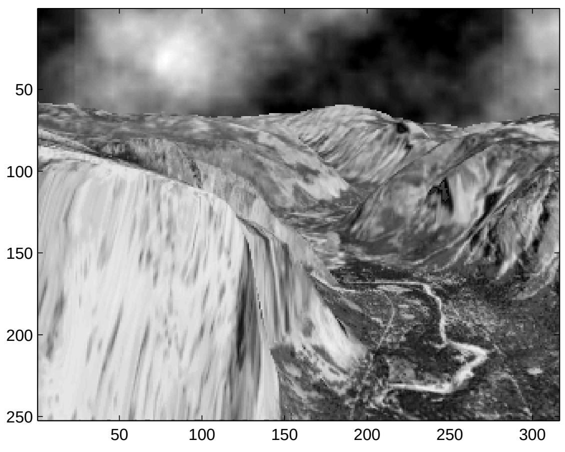


Figure 14: The 4^{th} frame of the image sequence.

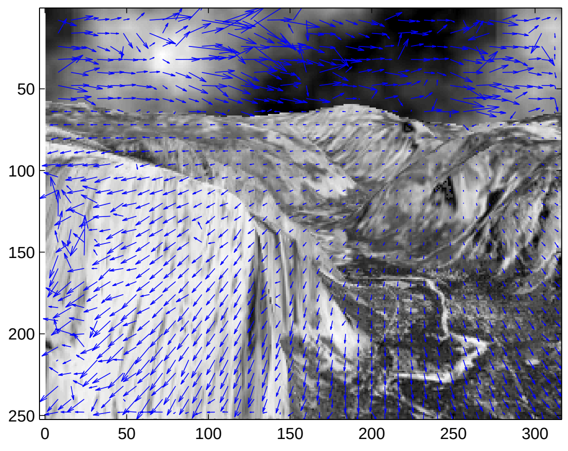


Figure 15: The optical flow field of the selected image sequence.

Unfortunately, the calculated optical flow and the true motion field are in general different, unless restrictions on both the lighting conditions and the objects being imaged are satisfied. As a result, this problem remains largely unsolved. We may further address this issue in our future works.

4.3 Motion Field Segmentation

4.3.1 Problem Definition

The optical flow of a pixel is a 2-dimensional vector indicating how this pixel moves in the scene. The next task for change detection or scene interpretation is to segment the data into homogeneous subsets such that each subset can be more easily represented or interpreted.

A traditional definition of segmentation is to choose a class of models with which to fit each subset. The typical approach to segmenting the data is to simultaneously decompose the mixture of all models into individual ones. Various methods have been proposed to resolve this problem, such as the K-means clustering algorithm and the expectation maximization (EM) algorithm.

In the problem of motion field segmentation, the number of segments is unknown. Therefore, determining the appropriate number of models for the data set becomes a more difficult problem. In order to resolve this issue, we will take a new approach based on minimum description length criterion. Before describing the algorithm, we will first give some further definition of this problem:

- The data $\Lambda \in \mathbb{R}^{M \times N}$ is a set of random samples from a mixture of models.
- The minimum description length criterion is given by

$$Min.Length(\Lambda, \theta) = Length(\Lambda|\theta) + Length(\theta). \tag{20}$$

- The optimal segmentation of the data Λ is a partition $\Lambda = \Lambda_1 \cup \Lambda_2 \ldots \cup \Lambda_N$ such that the overall coding length of the data Λ is minimal among all possible segmentations.

In general, the length function $Length(\theta)$ is chosen according to optimal lossless coding of a random source.

4.3.2 Coding Length based Segmentation

In this section, we will show how to calculate the expected coding length after segmenting the data into multiple groups. Suppose we already partitioned the set of data Λ into N non-overlapping groups:

$$\Lambda = \Lambda_1 \cup \Lambda_2 \cup \ldots \Lambda_N. \tag{21}$$

From [24, 25], we have the expected total number of bits required to encode the data Λ according to the above segmentation:

$$\begin{aligned} L^s(\Lambda, \Pi) &= \sum_{n=1}^{N} L(\Lambda_n) + |\Lambda_n|(-\log_2(|\Lambda_n|/M)) \\ &= \sum_{n=1}^{N} \frac{\text{tr}(\Pi_n + K)}{2} \log_2 \det(I + \frac{K}{\text{tr}(\Pi_n)\varepsilon^2} \Lambda \Pi_n \Lambda^T) \\ &\qquad + \text{tr}(\Pi_n)(-\log_2 \frac{\text{tr}(\Pi_n)}{M}), \end{aligned} \tag{22}$$

where the superscript s indicates the coding length after segmentation, and Π_i denotes the diagonal matrix that encodes the probability of M vectors in i:

$$\Pi_i = \begin{pmatrix} \pi_{1i} & 0 & \cdots & 0 \\ 0 & \pi_{2i} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \pi_{Mi} \end{pmatrix} \in \mathbb{R}^{M \times M}. \tag{23}$$

Here, π_{ji} denotes the probability that the *j*th vector belongs to the *i*th group.

Based on the above assumption, optimal segmentation is the approach that produces the minimal overall coding length $L^s(\Lambda, \Pi)$, and can equivalently be viewed as a good approximation to the actual entropy of the segmented data set.

An optimal solution to the coding length equation can be found in a bottom-up manner by merging regions of segments. A detailed proof is given in [24, 25].

4.3.3 Minimizing the Coding Length

We next seek to minimize the average coding length, given by

$$R^s(\Lambda, \Pi) = \frac{1}{M} L^s(\Lambda, \Pi), \tag{24}$$

as a function in Π.

Since the number of groups is unknown, we have to minimize $R^s(\Lambda, \Pi)$ over $N \in \mathbb{Z}_+$. From previous works, we know that any gradient-based descent algorithm relies on the initialization of data sets in order to converge to a global minimum. Since motion fields do not necessarily satisfy this requirement, it is quite difficult to minimize the rate function directly. Instead, we use a steepest-descent algorithm to minimize the length function $L^s(\Lambda, \Pi)$.

This algorithm is presented in Fig. 16. In each step, we choose two subsets of vectors, $\Lambda_i, \Lambda_j \in \Lambda$, such that by merging these two subsets, the decrement in the coding length is the largest. When the dimension of the space is relatively low, the greedy algorithm usually performs well. However, when the dimension of the subspace becomes high, the greedy algorithm does not always converge to the optimal solution.

4.4 Global Motion Estimation

In general, there are two kinds of motion, global motion and local or object motion. Global motion is caused by the movement of the camera and local motion is caused by the movement of objects.

In order to analyze the motion of the object, we have to first identify the global motion. This difficult task is similar to the problem of separating the background from the foreground in a static image. Some researchers have proposed approaches based on motion transformation and image rectification. However, these methods are computationally expensive and fail for motion fields in

Algorithm 1 **Steepest Descent Minimization**(Λ)

```
1. while flag == true do
2.     choose distinct sets $\Lambda_i, \Lambda_j \in \Lambda$ such that $L^s(\Lambda_i, \Lambda_j) - L^s(\Lambda_i \cup \Lambda_j)$ is maximum among all possible pairs
3.     if $L^s(\Lambda_i, \Lambda_j) - L^s(\Lambda_i \cup \Lambda_j) > 0$
4.         $\Lambda := (\Lambda \backslash \{\Lambda_i, \Lambda_j\}) \cup \{\Lambda_i \cup \Lambda_j\}$
5.     else
6.         flag = false;
7.     end
8. end while
9. return the set $\Lambda$
```

Figure 16: Steepest Descent Minimization Algorithm

the presence of inherent noise. In order to solve this problem, we propose a heuristic approach based on the properties of motion segments.

Generally, regions of global motion contain the corners of the scene. Although it is not always the case, however, we still could utilize this assumption combined with the variance statistics of the motion field segments to estimate the global motion. For each output segmentation, we will search for the motion region which contains the most corners of the scene. If there are more than one segments containing corners of the scene, the segment with the minimal variance of motion vectors will be selected to represent the global motion.

Once selected, the global motion is then taken as the average of all the motion vectors within the region. The algorithm is given in Fig. 17.

4.5 Change Detection

In this section, we describe our motion-segmentation based change detection algorithm. As shown in Fig. 18, the algorithm takes two images as input; the two images are captured by a camera at different time instants; the camera may experience some motion. Given the two images, in Step 1, Algorithm 2 **Global Motion Estimation**(Λ)

```
1. find the motion segment that contains the most number of corners
2. if more than one segment contains corners
3.     compare their variance of motion vectors
4.     return the segment that has minimal variance
5. else
6.     return the only segment
7. calculate the average of all the motion vectors in the segment, which is regarded as the global
   motion vector
```

Figure 17: Global Motion Estimation Algorithm

the algorithm solves the optical flow equation described in Section 4.2 and obtains a motion field. In Step 2, the motion field is segmented into regions with different motion by using Algorithm 1 described in Section 4.3. In Step 3, the global motion vector is estimated by Algorithm 2 described in Section 4.4. In Step 4, local motion vectors are obtained by removing global motion from the motion obtained in Step 2. Finally, the algorithm outputs 1) a global motion vector, 2) local motion vectors, and 3) regions of change. Note that regions of change are those that experience local motion.

4.6 Experimental Results

4.6.1 Motion Field Segmentation

To evaluate the proposed algorithm, we applied our segmentation approach to several video sequences containing various types of object motion.

Fig. 19 shows the first frame of an input image sequence containing two boats moving in a translational direction at different speeds along a river. To further complicate the experiment, global motion is introduced to the scene through the camera's movements. In Fig. 20, the optical flow field, computed using the algorithm in [20], is displayed. Finally, Figs. 21 through 24 show the four regions segmented by our proposed algorithm. With only some minor discrepancies between

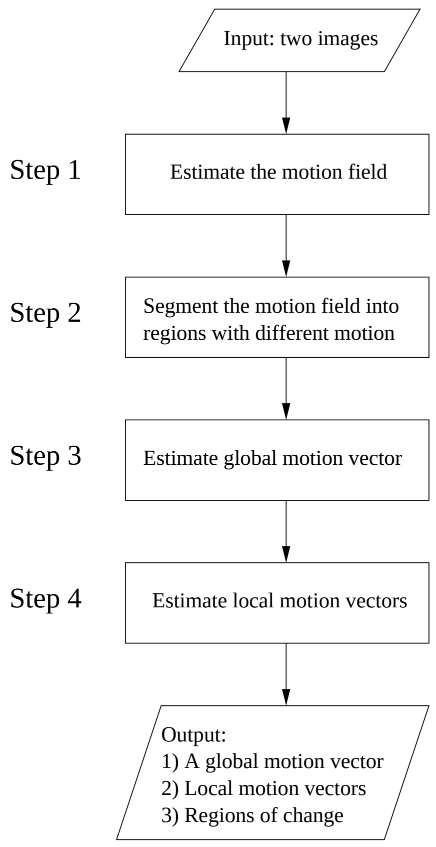

Input: two images
Step 1
Estimate the motion field
Step 2
Segment the motion field into regions with different motion
Step 3
Estimate global motion vector
Step 4
Estimate local motion vectors
Output:
1) A global motion vector
2) Local motion vectors
3) Regions of change

Figure 18: Flow chart of the algorithm for change detection.

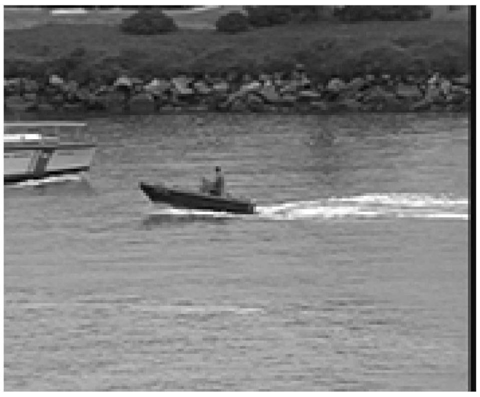


Figure 19: The 1^{st} frame in the boat image sequence.

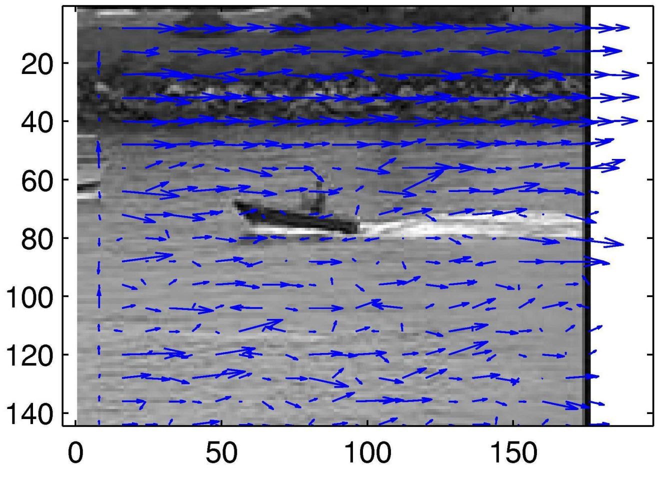


Figure 20: The computed optical flow field of the image sequence.

the land and water regions, each object in the sequence is successfully isolated by our approach.

4.6.2 Global Motion Estimation

We now seek to estimate the global motion in the boat sequence by applying our new, heuristic approach. The two segments containing the boats, seen in Figs. 21 and 22, do not contain any corners of the scene, and thus are not considered as candidates. The land and water segments, seen in Figs. 23 and 24, respectively, each contain two corners of the scene. As the variance of motion vectors is smaller in the land region, however, the vectors in this segment will be averaged to obtain an estimate of the global motion. As the land region is the only unmoving region, our method proves effective for this example.

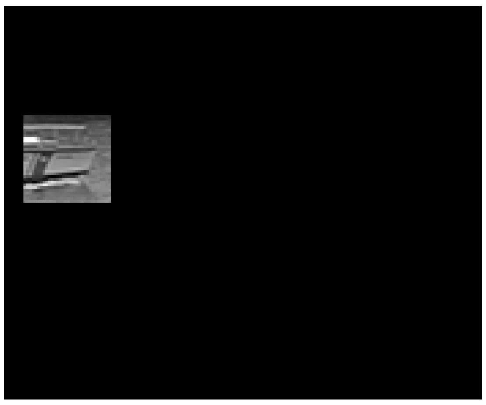


Figure 21: Segmented region from the image sequence containing one of the boats.

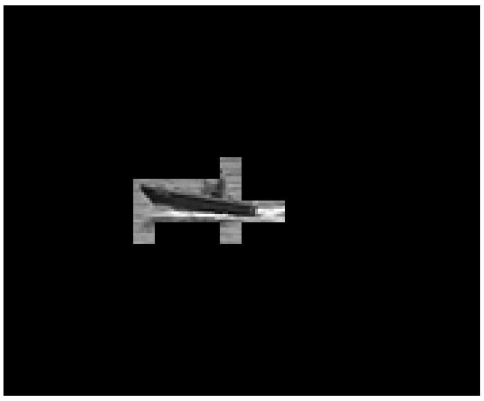


Figure 22: Segmented region from the image sequence containing one of the boats.

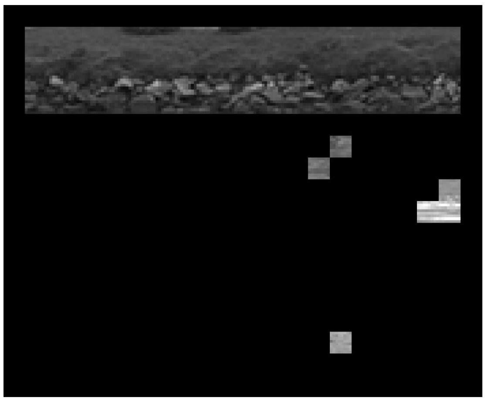


Figure 23: Segmented region from the image sequence containing the land.

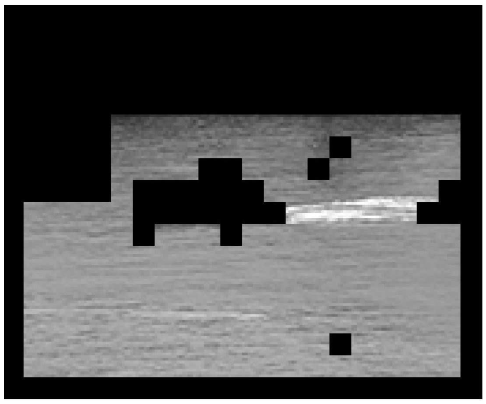


Figure 24: Segmented region from the image sequence containing the water.

5 Vehicle Tracking for Urban Surveillance

The objective of Task 3 is to devise feature-based algorithms to track vehicles in urban environments. The principal goal of tracking is to first identify regions of interest in a scene, and to then monitor the movements or changes of the object through the image sequence. In this project, we focus on unsupervised vehicle tracking for low resolution aerial images taken from an urban area. Various optical effects have traditionally made this tracking problem very challenging. Objects are often lost in tracking due to intensity changes that result from shadowed or partially occluded regions of an image. Additionally, the presence of multiple vehicles in a scene can lead to mistakes in tracking and significantly increased computation time. We propose a feature-based tracking algorithm herein that will seek to mitigate these limitations. To first isolate vehicles in the initial frame, we apply three-frame change detection to the registered images. Feature points are identified in the labelled regions using the Harris corner criteria. To track a feature point from one frame to the next, we search for the point around a predicted location, determined from the feature's previous motion, which minimizes the sum-of-squared-differences value. Finally, during the course of the image sequence, our algorithm constantly searches for new objects that might have entered the scene. Experimental results demonstrate the success of our tracking approach.

The results of Task 3 were published in Ref. [26]. Next, we present the technical details.

5.1 Introduction

Object tracking is a critical component in many computer vision applications, including medical imaging, surveillance, and robotics. Feature-based tracking, specifically, refers to the detection and monitoring of points, ideally located on objects of interest, through an image sequence.

The first step in any feature tracking algorithm is to identify a set of interesting points in an initial image. Tracking is then performed in subsequent frames by establishing correspondence between the current set of feature points and candidate points in the next image. Correspondence can be measured through various techniques, including correlation-based methods and sum-of-squared difference (SSD) determinations . For larger images, however, ambiguity often arises when multiple points display similar properties to the feature point of interest.

To prevent this ambiguity, and furthermore avoid the excessive time required to perform an exhaustive search, Lucas and Kanade presented an algorithm in Ref. 27 that instead uses the spatial intensity gradient of the image to direct the search for feature points. By attaining correspondence of features, they were able to register a set of images related by any linear transformation. In Ref. 28 , Tomasi and Kanade further generalized this algorithm to include feature-based tracking. For their approach, points were identified as features if they possessed good properties for tracking. During the tracking process, their algorithm monitors the residue of the feature, which refers to the point's change in appearance from the initial image to the current one. If the feature's residue becomes too large, it is declared no longer suitable for tracking. In this way, they again avoid potential errors in correspondence. However, in a realistic image sequence, the number of successfully tracked points would almost certainly decrease, as points could leave the scene or perhaps become affected by occlusion or lighting variations.

As an alternative to computing the image gradient, a Kalman filter [29, 30] could instead be used to direct the search for feature points. This approach uses knowledge of the point's motion through previous frames to predict where the point should lie in the current frame. After using the Kalman filter to smooth the object's trajectory, researchers often then use SSD measures to look for the feature point's best match in the directed area [31, 32]. A Kalman filter can also provide further benefit to a tracking algorithm. In Ref. 33, researchers use the Kalman filter to design an optimal template of the marked object, and thus help to detect possible occlusions that might occur in the video frames.

In this section, we will investigate the application of object tracking to vehicle and traffic surveillance. Several different approaches to this problem have already been presented by researchers. In Ref. 34, a feature-based tracking algorithm is described that uses a Kalman filter to direct the search for features. Detected feature points are then clustered together to identify vehicles in the scene. This approach requires the researcher to identify fiducial, or fixed, points in the images, and

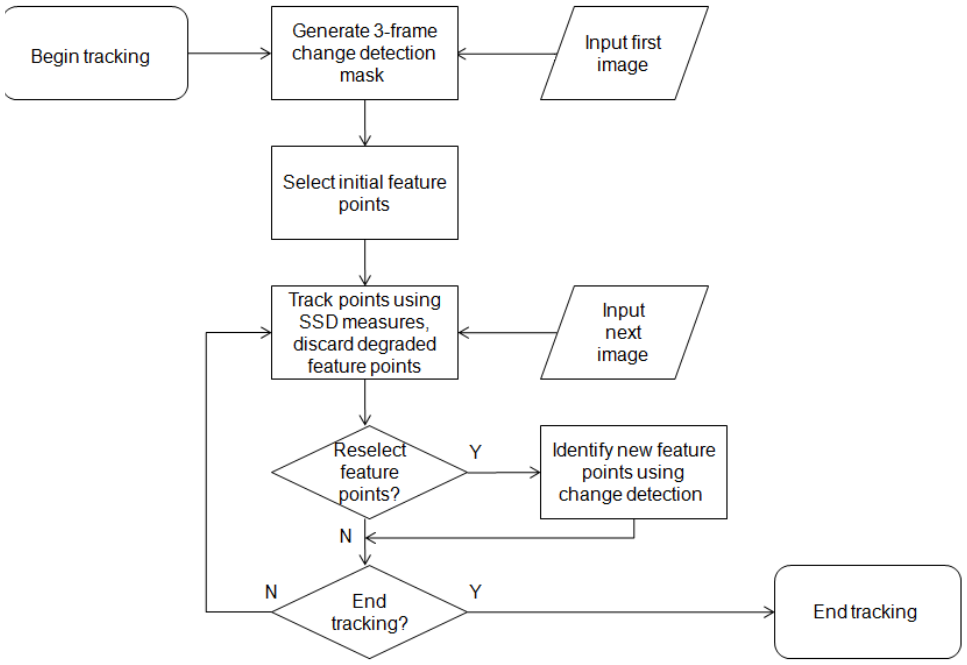

Begin tracking
Generate 3-frame change detection mask
Input first image
Select initial feature points
Track points using SSD measures, discard degraded feature points
Input next image
Reselect feature points?
Y
Identify new feature points using change detection
N
N
End tracking?
Y
End tracking

Figure 25: Algorithm flowchart.

furthermore demands high resolution images so that multiple points on an object can be marked. In contrast, researchers in Ref. 35 and Ref. 36 attempt to segment vehicles from the video sequence by creating a model for the image background. When significant motion is present in the scene, however, this type of segmentation approach would demand excess time in order to constantly update the background model. A similar approach is taken in Ref. 37, where objects are segmented using a block matching technique. However, this algorithm requires knowledge of the pixels' intensity changes over time, and demands a large amount of frames, and thus time, to create an accurate background model.

We will present herein a new approach to vehicle tracking that will serve to overcome the deficiencies in previous algorithms. We will adopt a feature-based tracking method that will specifically address the following problematic scenarios:

- A vehicle changes direction.
- Two vehicles pass each other.
- A large number of vehicles are present in the scene.

- A vehicle becomes temporarily occluded.

The algorithm will be capable of tracking vehicles even for low resolution images, and will also seek to minimize computation time, a necessary demand in many realistic applications.

The remainder of this section is organized as follows. In Section 5.2, we describe our approach to feature-based vehicle tracking. In Section 5.3, the performance of the algorithm is tested on various situational video sequences.

5.2 Tracking Algorithm

In this section, we will describe the different components of our feature-based tracking algorithm. The algorithm is summarized in the flowchart shown in Fig. 25.

5.2.1 Feature Selection

The first step in any feature-based tracking algorithm is to identify a set of candidate points in the initial frame of a video sequence. We measure the quality of a point using the Harris corner criteria, which is essentially a measure of the intensity variation in a neighborhood around the pixel. If the relative coordinates of the pixel are given by $\mathbf{x} = [x, y]^{\mathrm{T}}$, then the quality of the point is measured by

$$C(\mathbf{x}) = \det(\mathbf{G}) + k \times (\mathrm{trace}(\mathbf{G}))^2, \tag{25}$$

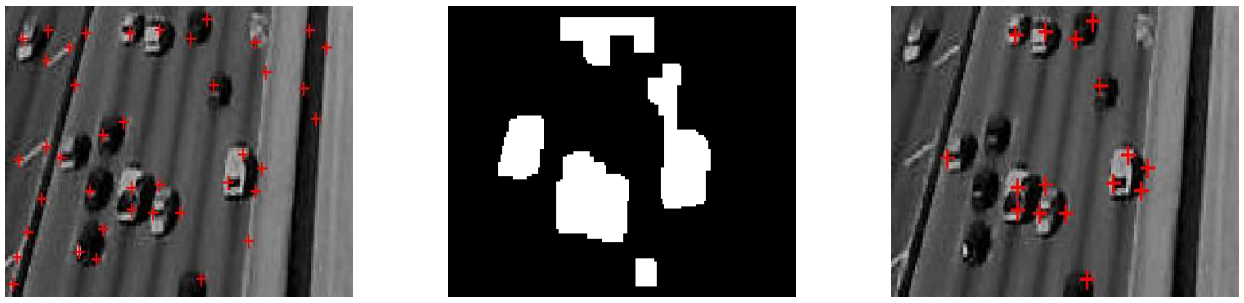
where $\mathbf{G}$ is a 2×2 matrix given by

$$\mathbf{G} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_x^2 \end{bmatrix}, \tag{26}$$

with I_x and I_y representing the horizontal and vertical intensity gradient, respectively, at the pixel of interest, and k is a scalar parameter determined by the user.

If the corner criteria exceeds a chosen threshold, then the point is selected as a chosen point for tracking. In image regions that contain a significant amount of intensity change, we further require a sufficient separation between tracking points so that the candidate feature list is sufficiently distributed in the image.

After collecting a set of predominant feature points in an image, we then attempt to remove those points that are contained in the background and thus only retain those pixels that lie on



(a) (b) (c)

Figure 26: (a) Image with marked feature points. (b) Binary object mask. (c) Image with feature points after application of mask.

vehicles in the frame. To isolate the feature points of interest, we use three-frame change detection, or image differencing, to generate a binary mask. Thus, points will be selected from only those regions in the image where motion has occurred in the previous three frames. We further smooth and denoise the mask using a basic set of morphological filters, whose parameters are determined from the average size of vehicles in the image sequence. The feature selection process is illustrated in Fig. 26.

5.2.2 Feature Tracking

After obtaining a set of feature points in the initial frame, we then wish to track the movement of those points over subsequent frames. To track a single point feature, an algorithm must identify a region of possible matches with which to search, and furthermore adopt a criteria to measure the quality of each possible match.

In our approach, we will use the previous motion of the feature point to predict where the point will lie in later images. For the second image, no motion history is available to the algorithm. Thus, a somewhat large search space will be defined around the target's previous location, as the movement of the feature could occur in any direction. After a match is found in the second frame, a motion vector is then defined by differencing the location of the feature in the first frame to its location in the second frame. In this way, the location of the feature in the third image is estimated, so that only a smaller, refined search space is needed. This process is continued for all successive images. By shrinking the search space, computation time and possibilities of a false match are both reduced. Although the movement of the feature point might change from one frame to the next (for example, as a vehicle changes direction), we assume that the frame rate is sufficiently high so

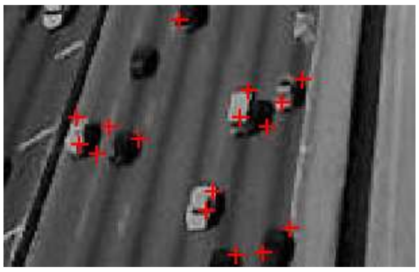


Figure 27: Image 50 in highway scene containing multiple vehicles.

that the point's motion is approximately constant for temporally adjacent images.

To determine the similarity between a feature point and a possible match, we use the sum-of-squared-differences (SSD) criteria. If a feature point on an image occurring at time t is located at $\mathbf{x} = [x, y]^{\mathrm{T}}$, then the intensity values of a neighborhood of pixels centered at $\mathbf{x}$ is given by $\mathbf{I}(\mathbf{x}, t)$. The best match for the feature point in an image taken at time $t + \delta t$ is the point located at $\mathbf{x} + \mathbf{\Delta}$, where $\mathbf{\Delta} = [\delta x, \delta y]^{\mathrm{T}}$ is the displacement vector that minimizes the SSD measure:

$$E(\delta x, \delta y, t) = \left[\mathbf{I}(\mathbf{x} + \mathbf{\Delta}, t + \delta t) - \mathbf{I}(x, y, t)\right]^2 . \tag{27}$$

If the SSD value of the best possible match for a feature point is still greater than some user-defined threshold, then the feature is declared no longer suitable for tracking.

5.2.3 Feature Reselection

Various types of occlusion can cause features to be temporarily lost throughout the course of a track. We have implemented a feature reselection option that allows us to search for and recover features that may have been lost. Further, by periodically searching for new feature points, we are able to detect objects that were not present in the initial scene, or perhaps were not moving in the initial scene (as image differencing is used to generate the detection mask).

Feature reselect uses the same principles as the methods that are used to identify features in the initial frame. After user-determined intervals, we calculate the number of features that have been lost throughout that interval. We then attempt to recover as many features as we lost by creating a mask that makes points of interest more prevalent, then applying Harris corner detector. However, the original quality thresholds still hold, meaning that not all features must be recovered during the reselection process. This accommodates points that have moved out of the frame, which is common when tracking moving features.

5.3 Numerical Results

In this section, we present several examples to demonstrate the success of our algorithm. Although our approach is mostly unsupervised, there are a few variables that must be set by the user. All parameters are dependent on the proximity of the camera to the scene, and the resolution of the camera.

As mentioned, a point is identified as a feature based on its quality, as measured in Equation (25). For each video sequence, a window size (in pixels) of 5×5 was used to determine a point's corner criteria. In contrast, a 7×7 window was used for all SSD measurements in the tracking phase. To prevent erroneous results around the edges of image frames, an appropriately sized boundary was marked out of consideration. Finally, the researcher also chose appropriate thresholds for feature selection and feature tracking, which was determined, again, based on the size of vehicles in the video.

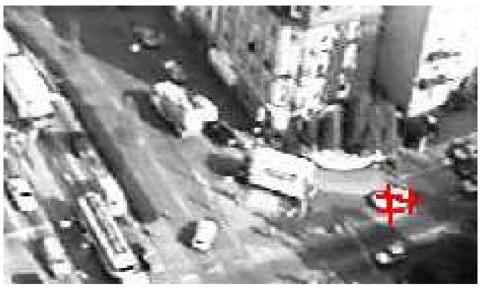
5.3.1 Multiple vehicles

We first consider a video containing multiple vehicles moving in the scene, as we hope to demonstrate that the algorithm is not limited by the number of cars present. The video is captured from just above a highway, and shows vehicles travelling in a linear path across the images. Vehicles continuously enter and leave the scene, and up to 12 cars are present at one time.

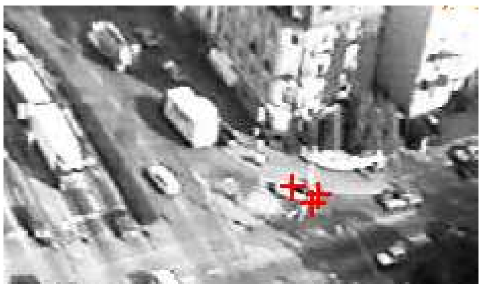
Frame 50 in the 75 frame video sequence is shown in Fig. 27, with feature points denoted by the red plus signs. For this image, 8 out of 9 vehicles in the scene are identified by feature points; this result was similar for other frames in the sequence. The algorithm is able to track almost all of the cars, including those that enter the scene during the video. However, as in the image shown, the algorithm has some difficulty tracking the darker, slower moving vehicles, which only start moving about halfway through the video.

5.3.2 Turning vehicle

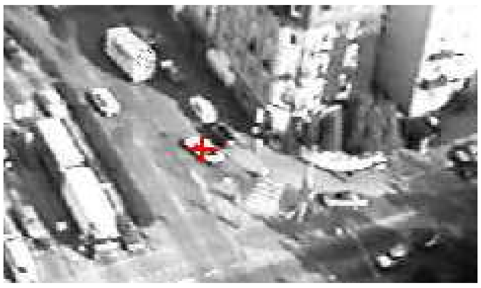
In the next example, we consider the ability of our algorithm to track a car as it changes direction. Specifically, we use a traffic video containing a vehicle that makes a right turn. For this sequence,



(a)



(b)



(c)

Figure 28: Turning vehicle sequence: (a) Frame 1. (b) Frame 11. (c) Frame 21.

we have applied an additional mask to isolate the vehicle of interest and remove uninvolved feature points. Furthermore, we do not apply feature reselection, but only select features in the initial frame.

In Fig. 28, we show frames 1, 11, and 21 of the video sequence, which contain the vehicle before, during, and after the turn, respectively. As shown, the car is successfully tracked throughout the maneuver. We also applied the KLT algorithm to this video. However, all feature points were lost as soon as the vehicle began to change direction. All the tracking parameters were held constant for both algorithms.

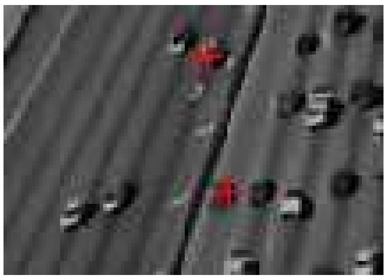
5.3.3 Passing vehicles

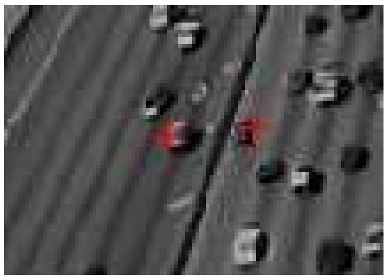
The third video sequence contains two vehicles that pass each other travelling in opposite directions on a highway. We thus hope to show that the algorithm is able to track multiple vehicles, independent of their proximity and direction of motion. As in the previous example, we have applied a mask to this sequence isolating the cars of interest. To demonstrate the effectiveness of our approach, we do not apply feature reselection, although this technique would certainly further enhance our results.

Frame 1, 6, and 11 of the video sequence are shown in Fig. 29, which show the cars before, during, and after they pass each other. Both cars are successfully tracked throughout the video sequence. When we applied our algorithm to scenes involving more vehicles (again travelling in different directions relative to each other), the results were consistent with this example. We also applied the KLT algorithm to this sequence; the results were similar to the ones obtained with our algorithm.

5.3.4 Reappearing vehicle

Finally, we highlight the effectiveness of our feature reselection process by considering a video sequence containing a vehicle that undergoes temporary occlusion. In Fig. 30, we show a vehicle before, during, and after partial occlusion by a stationary object in the scene. Several points are initially identified on the object, but all feature points on the vehicle are lost when the vehicle crosses behind a road sign. As the car reappears in the third frame, using feature reselection, features on the vehicle are recovered and it is reestablished as a point of interest.


(a)


(b)

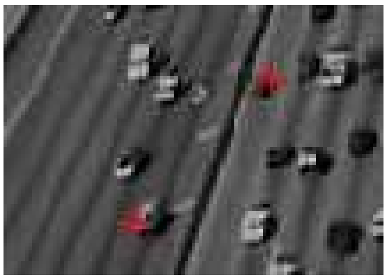
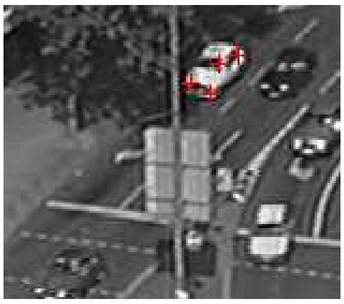
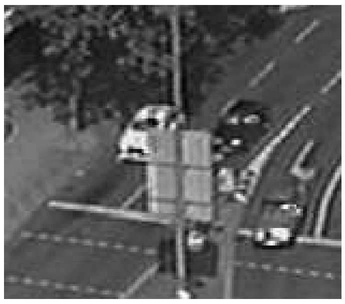

(c)

Figure 29: Passing vehicle sequence: (a) Frame 1. (b) Frame 6. (c) Frame 11.


(a)


(b)

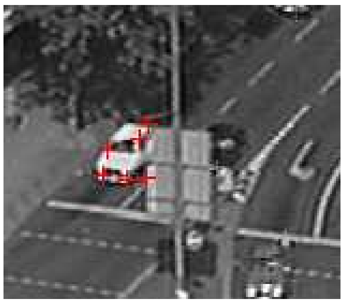

(c)

Figure 30: Occluded vehicle sequence: (a) Frame 1. (b) Frame 30. (c) Frame 50.

6 A Target Detection Scheme for VHF SAR Ground Surveillance

The objective of Task 4 is to design supervised learning algorithms for change detection and test our algorithms using the VHF change detection problem set.

Detection of targets concealed in foliage is a challenging problem and is critical for ground surveillance. To detect foliage-concealed targets, we need to address two major challenges, namely, 1) how to remotely acquire information that contains important features of foliage-concealed targets, and 2) how to distinguish targets from background and clutter. Synthetic aperture radar operated in low VHF-band has shown very good penetration capability in the forest environment, and hence the first problem can be satisfactorily addressed. The second problem is the focus of this paper. Existing detection schemes can achieve good detection performance but at the cost of high false alarm rate. To address the limitation of the existing schemes, in this project, we develop a target detection algorithm based on a supervised learning technique that maximizes the margin between two classes, i.e., the target class and the non-target class. Specifically, our target detection algorithm consists of 1) image differencing, 2) maximum-margin classifier, and 3) diversity combining. The image differencing is to enhance and highlight the targets so that the targets are more distinguishable from the background. The maximum-margin classifier is based on a recently developed feature weighting technique called I-RELIEF; the objective of the maximum-margin classifier is to achieve robustness against uncertainties and clutter. The diversity combining utilizes multiple images to further improve the performance of detection, and hence it is a type of multi-pass change detection. We evaluate the performance of our proposed detection algorithm, using the SAR image data collected by Swedish CARABAS-II systems which operates at low VHF-band around 20-90 MHz. The experimental results demonstrate superior performance of our algorithm, compared to the benchmark algorithm associated with the CARABAS-II SAR image data. For example, for the same level of target detection probability, our algorithm only produces 11 false alarms while the benchmark algorithm produces 86 false alarms.

The results of Task 4 were published in Ref. [38].

Next, we present the technical details.

6.1 Introduction

Detecting targets concealed in foliage or camouflage in a large area is a challenge problem and is critical for ground surveillance. It has many applications such as detecting the deployment of enemies hidden in the forest, locating the position of an accident in forest rescue activities and marking the foliage-covered terrain changes.

There are mainly two difficulties for this problem, namely, 1) how to remotely acquire information that contains important features of foliage-concealed targets, and 2) how to distinguish targets from background and clutter. Synthetic aperture radar (SAR) operated at low VHF-band is a good solution to the first problem [39][40]. At the low VHF-band around 20MHz - 90MHz, radar wave is more likely to make a return on objects exceeding certain dimension. Since this dimension is usually much larger than the leaves and branches under which targets are concealed, radar signal can penetrate the forest canopy and get reflected by the targets under it. These backscatters describe the scene covered and are used to form SAR images in which large objects show themselves as bright areas. In this way, VHF-band SAR technology transforms the foliage penetration problem into a traditional image based target detection problem. The second problem is also known as automatic target detection (ATD) problem. Major techniques for ATD include adaptive boosting [41], extended fractal feature [42], genetic programming [43], multiscale autoregressive (MAR), multiscale autoregressive moving average (MARMA) models, singular value decomposition (SVD) methods [44] and constant false alarm rate (CFAR) processing [45]. CFAR processing is widely used to give a globally applicable threshold for a constant probability of false alarms through estimating and removing the local background statistics.

According to Lundberg et al. [46], the main technical challenge in designing an ATD algorithm lies in how to keep the false alarm rate at a low level while yet achieving high detection rate. Suppressing false alarms is especially important in the case of detecting concealed targets because the foliage can add substantial amount of noise to the image. The denser the forest is, the noisier the image looks. In certain cases, it is almost impossible to distinguish targets from noise and background clutter such as huge trunks and rocks, given a single image. To mitigate this problem, multiple images can be used to suppress noise and background clutter.

For ground surveillance, it is reasonable to assume that targets are in the areas of change between two images taken at different times while background clutter is unchanged between the two images. Hence, the target detection problem can be addressed by change detection techniques, if multiple images are given. The objective of change detection is to find areas of change between

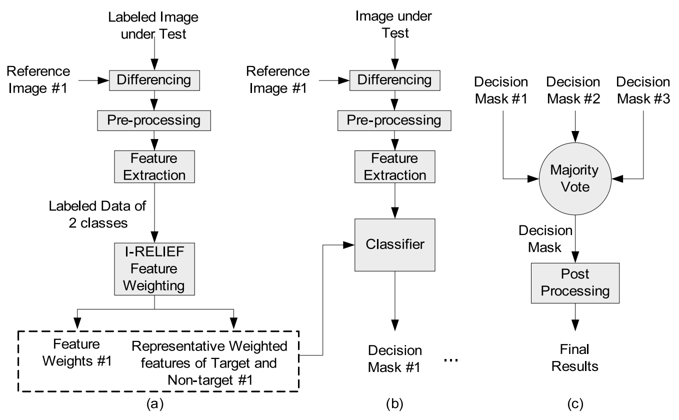


Figure 31: Flow chart: (a) learning; (b) testing; (c) diversity combining

an image under test and a reference image; here, an image under test may contain targets, and a reference image is an image of the same geographical region as the image under test; a reference image is captured at a different time and may not contain targets or contain targets that are in totally different locations, compared to the image under test. The changed areas or differences between two images may contain targets of interest, and the differencing can greatly suppress the background noise and clutter. Hence, change detection can help increase the probability of detection and reduce the false alarm rate [47][48][49][50].

In this section, we propose a target detection scheme that leverages change detection and a max-margin classifier based on nonlinear I-RELIEF feature weighting technique. In our target detection scheme, change detection helps separate targets from static background; the max-margin classifier makes our algorithm more robust to noise and unexpected clutter; we also use diversity combining to boost the performance of the algorithm further. Specifically, the first step is image differencing between an image under test and a reference image. Here, we assume that all the images are geometrically registered so that the same pixel in two images corresponds to the same geographical location; and all the images are radiometrically adjusted so that the lighting variation between two images is removed. Then, feature extraction, feature weighting and distance-ratio-based classification are applied to the difference image. Refer to Section 6.4 for more details. Diversity combining means combining signals from multiple sources into a single improved signal. In our case, differencing images between a specific image under test and multiple independent reference images are considered as diversity sources. After the same classification process on each source, multiple decisions are obtained independently. Majority voting among all the decisions gives a final decision, which is more reliable. Of course, the number of sources needs to be odd as required by majority voting. Fig. 31 shows the flow chart of our proposed algorithm.

We evaluate our proposed algorithm using a public released data set [51] acquired with the airborne CARABAS-II system which produces SAR images at low VHF-band around 20MHz – 90MHz during a flight campaign held in northern Sweden. It includes 24 images with 4 different targets deployments and 6 different flight passes for each deployment [46]. The image corresponding to mission 3 pass 5 was used for learning and the rest 23 images were used for testing. With different threshold values for distance ratio in the classifier, a performance curve of correct detection rate versus number of false alarms is obtained. Compared with the benchmark algorithm associated with the data set [46], our proposed scheme achieves much lower false alarm rate while yet achieving the same target-detection probability.

The remainder of the section is organized as below. In Section 6.2, we describe the image data set used for target detection. In Section 6.3, a baseline algorithm associated with the data set is presented. Section 6.4 presents our proposed scheme, which consists of four parts, namely, feature extraction, feature weighting, classification, and post processing. Section 6.5 shows our experimental results and compare our proposed scheme with the baseline algorithm.

6.2 Data Description

The image data set used for the evaluation of our proposed target-detection scheme is a subset of data collected during a flight campaign held in Sweden in the early summer of 2002 [51]. The images were taken by CARABAS-II, the second generation ultra-wide-band SAR mounted on a Sabreliner airplane. The system was operated in the frequency range of 20MHz - 90MHz. The corresponding wave lengths are between 3.3 meters and 15 meters. This dimension is much larger than that of leaves and branches and close to dimensions of vehicles which are the imagined targets to be pursued.

In this data set, there are several disturbing factors such as heading difference between images, different target orientations, different target sizes and radio frequency interference. The campaign was run at a spot in northern Sweden which mainly include a river and two blocks of forests. A rectangular area of 3km by 2km is chosen as the focus of all the data. And the GPS parameters of the corners are given with the data set. The recorded SAR images are all in 3000 by 2000 pixel size. Each pixel in the image corresponds to an area of 1m by 1m on the ground. 25 vehicles of three types and four deployments are used as targets hiding in the forests. There are ten TGB11 ($4.4 \times 1.9 \times 2.2$ m), eight TGB30 ($6.8 \times 2.5 \times 3.0$ m) and seven TGB40 ($7.8 \times 2.5 \times 3.0$ m). TGB11, TGB30 and TGB40 represent different kinds of vehicles. Sigismund, Karl, Fredrik and Adolf-Fredrik are

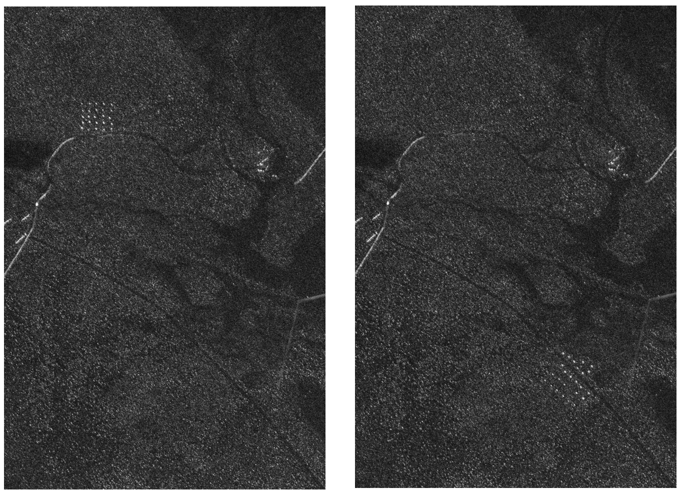


Figure 32: Two sample images from the data set. Left: Sigismund deployment, flight heading 225; Right: Fredrik deployment, flight heading 225

the codes of the four deployments. The vehicles have totally different positions and orientations between different deployments. To guarantee the variety of the data set, for every single vehicle deployment, six different flight headings were adopted. So, totally 24 images were acquired. There is a TV transmitter which is the source of radio frequency interference located south-east of the focused area. This affects the SAR imaging process and makes different contributions for different flight heading angles. Along with the data set, the actual position of each vehicle is given in the form of GPS parameters for the purpose of evaluation. See Fig. 32 for example images. Please refer to [46] for more information of the data set.

6.3 Baseline Algorithm for Target Detection

Along with the data set, a baseline algorithm and experimental results are given in [46]. Details about the algorithm are discussed in [52]. This algorithm is based on change analysis, statistic hypothesis test and CFAR normalization techniques. First of all, two pixel values are extracted from image under test I_t and reference image I_r at the corresponding position respectively. They are combined into a vector as:

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{28}$$

It is assumed that each pixel value in the image is a random variable and values of different pixels are statistically independent. z_1, z_2 are two real-valued random variables and $\mathbf{z}$ is a random vector.

Then, target or change, clutter and noise signal are defined in the similar way as:

$$\mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}, \ \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \ \mathbf{n} = \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}, \tag{29}$$

where indices 1 and 2 indicate the two images. Hypothesis test is used to determine whether a change is present. The two hypotheses are defined as:

$$\begin{array}{ll} H_0 : \mathbf{z} = \mathbf{c} + \mathbf{n} = \mathbf{q} & \text{(no change)} \\ H_1 : \mathbf{z} = \mathbf{s} + \mathbf{c} + \mathbf{n} = \mathbf{s} + \mathbf{q} & \text{(change)} \end{array} \tag{30}$$

A test statistic of likelihood ratio is computed to test the two hypotheses. The test statistic is defined by:

$$\Delta(\mathbf{z}) = \frac{P(\mathbf{z}|H_1)}{P(\mathbf{z}|H_0)} \tag{31}$$

The random vectors of noise **n** and clutter **c** are assumed to have bivariate zero-mean circular Gaussian distribution. So the sum of the two **q** also follows bivariate zero-mean circular Gaussian distribution. The two probability density functions (PDF) of the likelihoods in (31) are:

$$P(\mathbf{z}|H_0) = \frac{1}{\pi^2|\mathbf{C}|}\mathbf{exp}(-\mathbf{z}^T\mathbf{C}^{-1}\mathbf{z}) \tag{32}$$

and

$$P(\mathbf{z}|H_1) = \frac{1}{\pi^2|\mathbf{C}|}\mathbf{exp}(-(\mathbf{z}-\mathbf{s})^T\mathbf{C}^{-1}(\mathbf{z}-\mathbf{s})) \tag{33}$$

where **C** is a 2×2 covariance matrix defined by:

$$\begin{aligned} \mathbf{C} &= E\{(\mathbf{z} - E\{\mathbf{z}\})(\mathbf{z} - E\{\mathbf{z}\})^T\} \\ &= \begin{pmatrix} \sigma_1^2 & \rho_{21}\sigma_2\sigma_1 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \end{aligned} \tag{34}$$

where $\sigma_k^2 = E\{q_k^2\}$ and $\rho_{kl} = E\{q_k q_l\}/(\sigma_k\sigma_l)$. The value of covariance matrix **C** is estimated using pixel values within smaller local image blocks. Then the statistic test is

$$\begin{aligned} |\mathbf{s}^T\mathbf{C}^{-1}\mathbf{z}| > \lambda &\ \rightarrow \text{decide } H_1 \\ |\mathbf{s}^T\mathbf{C}^{-1}\mathbf{z}| \leq \lambda &\ \rightarrow \text{decide } H_0 \end{aligned} \tag{35}$$

It is assumed that the target is not present in one image but present in the other image, that means:

$$\mathbf{s} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \text{ or } \mathbf{s} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{36}$$

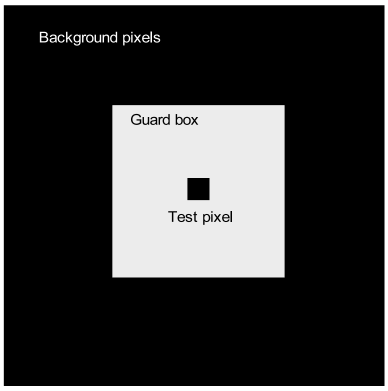


Figure 33: The area used for computing the background statistics surrounding a pixel under test

With the change detection algorithm described above, we can convert two SAR images into one binary image with ‘0’ indicating no change and ‘1’ indicating change. Next, we describe how to determine the value of threshold λ. Because the statistic features vary at different locations in an image, the threshold λ should also change for different locations. This change is difficult to predict for different images. To normalize the value of threshold λ, a CFAR detection filter is used. It is used to estimate the local background statistics in the change image and remove the variance between different locations. The shape of the CFAR filter is shown in Fig. 33. The background statistics, mean value and standard deviation are estimated based on the pixel values within the outer box but outside the inner box. The sizes of outer box and inner box are 31×31 and 19×19 respectively. Then the background statistics are compared with the pixel under test. The mean value is subtracted from the center pixel, then divided by the standard deviation. The resulting value is compared with a global threshold λ' to make a decision. After thresholding, morphological operations, i.e., one erosion operation and two dilation operations are applied to the binary change mask to remove false alarms.

6.4 Our Proposed Scheme for Target Detection

The flow chart of our proposed target-detection scheme is shown in Fig. 31. The input of our algorithm is SAR images and the output of our algorithm is locations of possible targets, if there is any. In the situation considered in this section, as in most situations, a target occupies an area of more than one pixel in an image. But the location of a target is usually specified by the coordinates of a single point, e.g., the centroid, the top-left corner point, or the very top point of the target. So the ATD task is divided into two subtasks: 1) label each pixel as target or non-target; 2) group connected target pixels into targets and extract their coordinates. Usually, the first part

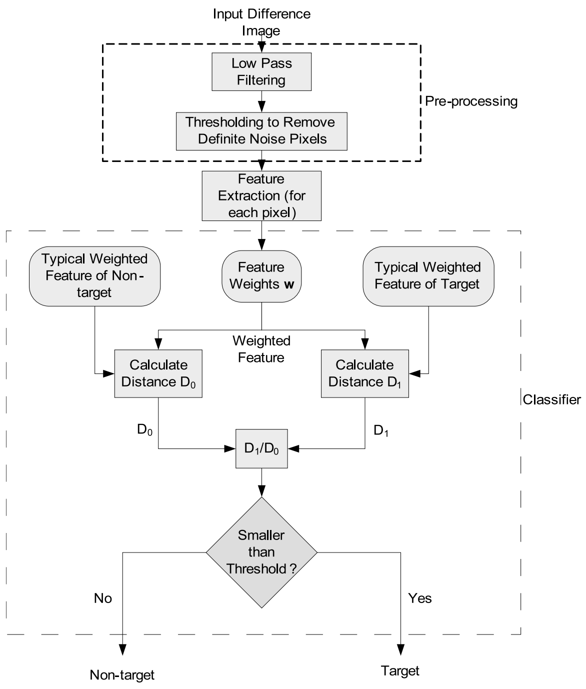


Figure 34: Flow chart of our classification algorithm

is much more challenging than the second part. If the image is correctly labeled, the second step becomes simple. The first subtask is actually classification. A carefully designed classifier is the core of this subtask. In our scheme, we use a distance-ratio-based classifier and train the classifier through supervised learning. Fig. 34 shows the flow chart of our proposed classification algorithm. The major steps of our classification algorithm are feature extraction, feature weighting, and classification/target-detection, which are presented in the following sections, respectively.

6.4.1 Feature Extraction

Our feature extraction method is based on change analysis. The first step is to acquire change information. Simple differencing between image under test I_t and reference image I_r is adopted here. A difference image I_d is obtained by

$$I_d = \begin{cases} I_t - I_r & I_t - I_r > 0 \\ 0 & I_t - I_r \leq 0 \end{cases} \tag{37}$$

In the difference images, target areas should always be brighter than background areas. All the images in this data set are all geometrically registered. In order to reduce the influence of noise on classification, the difference image is processed by a low pass filter. It is equivalent to be

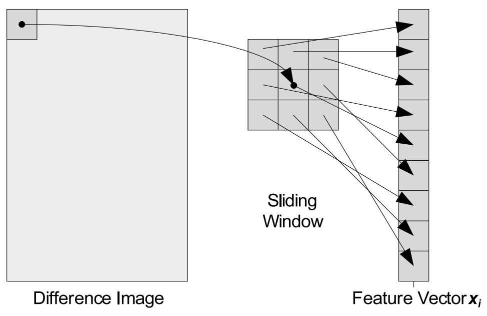


Figure 35: Feature extraction process

convoluted by a core matrix with averaging effect. A convolution core of 5×5 is used. Also, a pre-screening step is performed. It forces those pixels with intensities lower than a given threshold to be 0. This threshold should be enough to preserve all the target pixels. It is intended to remove those pixels that are obviously noise or background. These denoising processes can contribute to boosting the performance of the classifier and increasing the converging speed at the training stage. All the input data is SAR images of 3000 by 2000 pixels. Because there are multiple targets in each image, it is not reasonable to take the whole image as a sample and label it as target or non-target. Instead, for each pixel, a local feature set is extracted and conveyed to the classifier. Then, each pixel is classified as target pixel or non-target pixel. In the baseline algorithm, only the intensity of current pixel is conveyed to the classifier as features. It does not consider the local environment of that pixel and is easy to be fooled by some high-intensity noise. To avoid this problem, for every pixel in the SAR image, all its neighbors within a $2n + 1 \times 2n + 1$ window are considered helpful in representing the behavior of that pixel and used to extend the feature extracted from the current pixel. Their pixel values, Fourier transform coefficients or other transform results can all be used as features. These features include lots of local information in addition to one pixel value and are widely used in image analysis [53]. Here, for a given pixel i, original pixel values of its neighbors are extracted and reordered to form a feature vector $\mathbf{x}_i$. Then, by sliding a window within the image, a set of features X is obtained. Fig. 35 shows the process of feature extraction.

6.4.2 Feature Weighting

As mentioned in the previous section, all the pixel values within the neighborhood of pixel under test are extracted to form a feature vector $\mathbf{x}$. These features do not have the same contribution to discriminating the target and non-target classes. For example, the feature pixels closer to the pixel under test, i.e., near the center of the sliding window, deserve more attention than those farther away from the pixel under test, i.e., near the border. Otherwise, in a non-target case, the noise within the border area may trigger the classifier to decide that a target is present. In the training stage, some features which are very noisy may prevent the training algorithm from converging or lead to the classifier's over-fitting to the training data. In order to figure out how much attention should be paid to each feature, feature weighting, which assigns a real-valued number to each feature is adopted. The real-valued number here is called the weight of the corresponding feature. For a given feature vector $\mathbf{x}$ and a global weight vector $\mathbf{w}$, a weighted feature vector $\mathbf{x}'$ is obtained by multiplying each feature $x^{(i)}$ with its corresponding weight $w^{(i)}$. Define

$$\mathbf{x} = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(I)} \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w^{(1)} \\ w^{(2)} \\ \vdots \\ w^{(I)} \end{bmatrix}, \quad \mathbf{x}' = \begin{bmatrix} x'^{(1)} \\ x'^{(2)} \\ \vdots \\ x'^{(I)} \end{bmatrix}, \tag{38}$$

where $x'^{(i)} = x^{(i)} \times w^{(i)},\ i = 1, 2, \ldots, I$, where I is the data dimensionality.

We use I-RELIEF or Iterative-RELIEF, an improved interpretation of RELIEF for feature weighting. The key idea of I-RELIEF is to solve a convex optimization problem with a margin-based objective function. For RELIEF, the margin is defined based on a 1-NN, i.e. one nearest neighbor classifier. It only considers one nearest neighbor in the same class and one nearest neighbor in the other class. For I-RELIEF, the margin is averaged between all the sample pairs weighted by the possibility of being an outlier. The feature weights are iteratively estimated according to their ability to discriminate between neighboring patterns. Most algorithms for feature weighting and feature selection, which is the specific case of feature weighting with weights taking values of 0 or 1, rely on heuristic searching, because it is hard to define an objective function that can be optimized with low computational complexity. So they do not guarantee to give optimal solutions. RELIEF [54] addresses this problem by optimizing an objective function with low computational complexity. Let $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_1^N \in \mathbb{R}^I \times \{\pm 1\}$ denotes a training data set, where I is the data dimensionality, $\mathbf{x}_n$ is a sample feature, y_n is the label of the feature, and N is the total number of samples in the training data set. Then, in RELIEF, the feature weighting problem is converted into the following optimization problem:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \textstyle\sum_{n=1}^{N} \left(\sum_{i=1}^{I} w^{(i)} |x_n^{(i)} - NM^{(i)}(x_n)| - \sum_{i=1}^{I} w^{(i)} |x_n^{(i)} - NH^{(i)}(x_n)| \right) \\ \text{s.t.} \quad & \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geq 0 \end{aligned} \tag{39}$$

where NM means nearest miss which is the nearest neighbor of x from different class and NH means nearest hit which is the nearest neighbor of x from the same class. This is generated from a natural idea of scaling each feature such that the averaged margin in a weighted feature space is maximized. The constraint $\|\mathbf{w}\|_2^2 = 1$ prevents the maximization from increasing to infinity. The constraint $\mathbf{w} \geq 0$ ensures the weight vector is a distance metric. I-RELIEF [55] is an improved version of RELIEF. It solves two major problems in RELIEF: 1) the nearest neighbors are defined in the original feature space, which may not be the nearest in the weighted feature space; 2) the margin calculation can be influenced by outliers greatly. Based on the assumption that nearest neighbors and identity of a pattern are hidden random variables, I-RELIEF iteratively estimates feature weights following the principle of EM algorithm until convergence. This algorithm is proved to be convergent [55].

6.4.3 Classification

From the training stage, an optimal feature weight vector $\mathbf{w}^*$ is generated. In the new weighted feature space, the distance between two different classes is maximized and the distance within the same class is minimized. In the weighted space, various distance-based classifiers can be designed. In this section, we choose a distance ratio to design our classifier. Next, we define this distance ratio.

In the training phase, we can obtain the centroid of the target sample set $\bar{\mathbf{x}}_{target} = \frac{1}{N_t}\sum_{i\in CT}\mathbf{x}'_i$ and the centroid of non-target sample set $\bar{\mathbf{x}}_{ntarget} = \frac{1}{N_n}\sum_{i\in CN}\mathbf{x}'_i$, where CT is the set of indices of target samples, CN is the set of indices of non-target samples, N_t is the number of target samples, and N_n is the number of non-target samples. Then, for a feature sample (under test) $\mathbf{x}$, the distance ratio is defined by

$$DR(\mathbf{x}) = \frac{|\mathbf{x} - \bar{\mathbf{x}}_{ntarget}|}{|\mathbf{x} - \bar{\mathbf{x}}_{target}|} \tag{40}$$

Given an input SAR image under test, a distance ratio is calculated for each pixel. Then, the distance ratio is compared to a threshold and a decision is made for each pixel, based on the following criterion:

$$\begin{aligned} DR(\mathbf{x}) > \lambda \quad &\rightarrow \quad \text{decide change/target, labeled as '1'} \\ DR(\mathbf{x}) \leq \lambda \quad &\rightarrow \quad \text{decide no change/non-target, labeled as '0'} \end{aligned} \tag{41}$$

The above process produces a binary-valued image with '1' denoting target and '0' denoting non-target. This image is also called change mask. For the same input SAR image under test, different reference images may result in different change masks. With multiple change masks, we can apply a majority voting rule to each pixel and obtain a final change mask. Note that for a different reference image, we need to re-do the training of our classifier since a different reference image represents a different training sample set.

6.4.4 Post Processing

The output produced by our classifier is a binary valued image. But the objective of target detection is to obtain the locations of targets. To achieve this, we group connected pixels whose value is “1”, and declare such a connected region as a target. The coordinates of the centroid of a connected region represents the location of the target associated with the region.

Another purpose of post processing is to remove false alarms. Since we have prior knowledge about the size of the targets, we can remove the connected regions whose sizes are smaller than an expected value. Our experimental results in the next section show the effectiveness of such post processing.

Processing step	Parameter	Value
Preprocessing	Averaging kernel size	5×5 pixels
	Denoising threshold	0.25
Feature extraction	Sliding window size	19×19 pixels
I-RELIEF feature weighting	Maximum number of iterations	500
	Distance metric	‘Euclidean’
	Kernel function	$f(d) = \exp(-d/\sigma)$
	Kernel width σ	25
Classification	Threshold λ on DR	1/3
Post processing	Minimum number of connected pixels as a target	35 pixels
Evaluation	Distance threshold	10 pixels

Table 1: Parameters used in the experiments

6.5 Experimental Results

In the experiments, for any given input image, the reference images are always chosen to be those images taken under the same flight angle but different deployments. The requirement of same flight angle ensures that the imaging conditions are the same. According to the data set, the four

Target image		Correct detections			False alarms		
Mission	Pass	Benchmark Algorithm	This Scheme 3 references	This Scheme 1 reference	Benchmark Algorithm	This Scheme 3 references	This Scheme 1 reference
2	1	25	25	25	2	0	0
3	1	22	24	23	1	3	3
4	1	25	25	25	2	0	0
5	1	23	25	25	4	2	3
2	2	25	25	25	2	0	0
3	2	25	25	25	4	1	1
4	2	25	25	25	3	2	1
5	2	25	21	21	4	0	0
2	3	25	25	25	3	1	1
3	3	23	23	23	4	0	0
4	3	25	25	25	0	2	1
5	3	24	25	25	2	0	0
2	4	24	25	25	3	0	0
3	4	25	25	25	2	0	0
4	4	25	25	25	4	0	0
5	4	25	22	23	4	0	1
2	5	25	25	25	3	0	0
3	5	(Used for training)					
4	5	25	25	25	2	0	0
5	5	23	25	23	29	0	10
2	6	25	25	25	1	0	0
3	6	25	24	24	3	0	0
4	6	25	25	25	1	0	0
5	6	23	25	25	3	0	0
Total		562	564	562	86	11	21

Table 2: Comparison of the benchmark algorithm, and our proposed scheme with majority vote from 3 references and with a single reference image.

deployments Sigismund, Karl, Fredrik and Adolf-Fredrik are denoted as mission 2, 3, 4 and 5 respectively. So, for each image, there can be no more than three different reference images from other deployments.

Table 1 shows the parameters used in our experiments. Our rationale of choosing 19×19 for the sliding window size is the following. For best performance, the sliding window should be large enough to cover a whole target. On the other hand, a larger sliding window means higher computational complexity. We tested sliding window sizes of 7×7, 19×19, 31×31 and 41×41 pixels. The later three achieve the same performance while the first one performs worse. Hence, we choose 19×19 for the sliding window size.

In the training stage, the image of mission 3 flight 5 is used as training samples. The coordinates of the 25 target vehicles are included in the data set. For each target, a sliding window is manually shifted within a 5×5 neighborhood of the given coordinates of the target, and one feature vector belongs to the target class is extracted from the sliding window at each position. A feature vector set labeled as target including totally 625 samples is extracted from the 25 target locations. Another feature vector set labeled as non-target is extracted from 625 background positions which are manually chosen to avoid the target region. These two label sample sets are used as the input of the I-RELIEF feature weighting algorithm to give an output of weight vector $\mathbf{w}^*$. Values of $\bar{\mathbf{x}}_{target}$ and $\bar{\mathbf{x}}_{ntarget}$ are also estimated from the training sample sets. According to the above reference look-up matrices, three different references are chosen, and training and testing are performed independently for each of the three references. Finally, a majority vote is applied to the three change masks obtained from the training and testing w.r.t. the three references, and the algorithm outputs the coordinates of all the detected targets.

For performance evaluation, the output coordinates are compared to the ground-truth target positions. If the location of a target detected by the ATD algorithm, is within the disk of 10-pixel (i.e., 10-meters) radius, centered at the ground truth position, then we declare that the detection is correct. If more than one target is found within this disk, one will be counted as a correct detection while others will be counted as false alarms.

Table 2 shows the testing results for 1) the benchmark algorithm [46], 2) our proposed scheme with majority vote from 3 references, and 3) our proposed scheme with a single reference image. In the testing stage, the image of mission 3 flight 5 is not used for testing because it serves as training samples. The experimental results show that compared to the benchmark algorithm [46], our proposed scheme with majority vote from 3 references produces much fewer false alarms, i.e.,

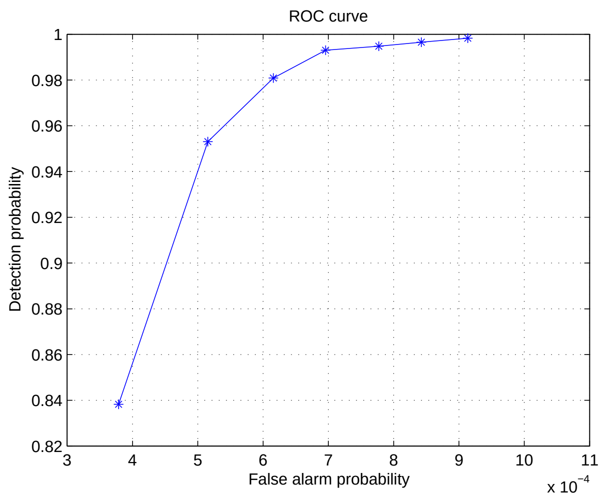


Figure 36: ROC curve

11 for ours vs. 86 for the benchmark, while yet achieving better target detection performance, i.e., 564 for ours vs. 562 for the benchmark. In addition, using diversity combining (multiple reference images) improves the performance of our proposed scheme with a single reference image.

The same experimental results for our proposed scheme with majority vote from 3 references in Table 2, can be presented by a receiver operating characteristic (ROC) curve, i.e., the false alarm probability vs. the detection probability as shown in Fig. 36. The false alarm probability in Fig. 36 is estimated at the pixel level, i.e., by the ratio of the number of false alarm pixels to the total number of pixels in an image; since the number of false alarm pixels is small and the total number of pixels in an image is very large (actually there are 6×10^6 pixels in an image), the false alarm probability is very small (in the order of 10^{-4}). The detection probability is estimated by the ratio of the number of correctly detected targets to the total number of targets. Since Ref. [46] does not provide an ROC curve, we do not plot the ROC curve for the benchmark algorithm. We were not able to implement the benchmark algorithm exactly since Ref. [46] does not provide detailed description of the morphological operations used in the benchmark algorithm.

7 A Rotation-Invariant Transform for Target Detection in SAR Images

The objective of Task 5 is to design rotation-invariant transform for change detection and test our algorithms using the VHF change detection problem set.

Rotation of targets poses a great challenge for the design of an automatic image-based target detection system. In this project, we propose a target detection algorithm that is robust to rotation of targets. Our key idea is to use rotation invariant features as the input for the classifier. In this work, for an image, its coefficients of the combined Radon and 1-D Fourier transform are proved to be rotation invariance. These coefficients are used as the input to a maximum-margin classifier based on I-RELIEF feature weighting technique. The objective of the I-RELIEF technique is to maximize the margin between two classes and improve the robustness of the classifier against uncertainties. For each pixel of the Synthetic Aperture Radar (SAR) image, a feature vector can be extracted from a sub image centered at that pixel. Then our maximum-margin classifier decides whether the pixel is target or non-target which produces a binary-valued image. We further improved the detection performance by connectivity analysis, image differencing, and diversity combining. Our performance evaluation of the proposed algorithm was based on the data set collected by Swedish CARABAS-II systems. In conclusion, the experimental results show that our proposed algorithm achieved superior performance over the benchmark algorithm.

The results of Task 5 were published in Ref. [56].

Next, we present the technical details.

7.1 Introduction

SAR imaging sensors can provide images of a wide ground region and has the ability to visualize what is being covered by the foliage [46][52]. At the low VHF-band, around 20MHz - 90MHz, radar waves are more likely to detect targets that exceed a certain dimension. Since this dimension is usually much larger than the leaves and branches, the sensors are able to detect the concealed objects underneath the forest. The reflected radar waves from the hidden objects are used to form SAR images in which the larger targets are seen as brighter areas than the smaller objects.

Another problem is to develop an algorithm that can automatically analyze the image and provide the essential information. For example, the essential information can be scene type, existence of certain objects, or location of all the specified objects. In this section, all the research is made based on a data set captured by CARABAS-II radar which can be downloaded at [51] for free. The purpose of this project is to locate all the vehicles concealed in the forest which is known as automatic target detection (ATD). Also a benchmark algorithm has been provided with the dataset by [46], but this algorithm detected too many false alarms; therefore, further research needs to be conducted to develop a more accurate algorithm.

Techniques using adaptive boosting [41], extended fractal feature [42], genetic programming [43], multiscale autoregressive (MAR), multiscale autoregressive moving average (MARMA) models, singular value decomposition (SVD) methods [44], and constant false alarm rate (CFAR) processing [45] were studied. According to Lundberg et al. [46], the main technical challenge in designing an ATD algorithm for a forest covered region is not detecting targets, but reducing the false alarm rate to a useful level. The SAR is considered to be a good sensor in the foliage penetrating scenario; however, when the targets are concealed by the forest, the branches and leaves will cause a significant amount of noise to appear in the image. One thing to consider is the density of the forest and the noise in the image are directly proportional. Another key point is most algorithms work well for open areas, but not in the forest because of the strong noise which is produced by the leaves and branches of the trees.

One important assumption that is made when analyzing the images is that the background clutter is stationary and targets are non-stationary. Now due to this assumption that was made, the target detection problem is equivalent to the change analysis which can be defined as finding the differences between the test image and a reference image. The next important clarification is to define test image as an image in which the algorithm tries to locate the targets from the surrounding area. Also the reference image is an image of the same location as the test image, but taken when the targets have moved to a different location. In order to detect the moving targets in the test image, the algorithm takes the difference of the two photos and the outcome consist mostly of the moving targets. The effectiveness of the change-based ATD scheme has been proved by [47][48][49][50].

Along with the dataset [51], a benchmark algorithm and the results are given in [46]. This algorithm is a statistical hypothesis test followed by a CFAR filter and morphological post processing. In the statistical hypothesis test, the targets are assumed to be deterministic signal while the background clutter and noise are assumed to be Gaussian random variables. Then the decision is made for every pixel in the image according to Neyman-Pearson criterion. Now the statistical hypothesis test is used in the benchmark algorithm which is not the optimal solution because the statistical hypothesis test treats each pixel in the SAR image as independent random variable; therefore, loses the spatial information which is vital for detecting targets in the SAR image. Another reason why the statistical hypothesis test is not optimal is because this test uses the same statistical model for different targets.

In our previous work [57], we used a fundamentally different ATD algorithm which shows improvement in the results. Our past algorithm shares the same change analysis idea with the benchmark algorithm, but our scheme is able to determine the targets from local features of labeled SAR images. Now the previous algorithm leads to a more dedicated classifier for the particular target; however, our algorithm trades generality for performance.

Our framework for the new algorithm is similar to the previous algorithm [57] with the exception of a more advanced local descriptor. The function of the local descriptor is to extract local features from the given region of interest in the image. Then the local features computed for region of interest have been proved to be successful in applications of imagery data analysis [58]. In our previous work [57], local features are vectors whose elements are intensity values extracted from a sliding window centered at a pixel of interest. However, in this section, an extra rotation invariant transform step is applied to the region of interest. In order to receive the feature vectors which are invariant to different object orientations, we use the outcome of the extra rotation invariant transform as elements of the feature vectors. The diagram of the new algorithm is shown in Fig. 37.

Since all the learning and testing images are considered geometrically registered; therefore, the differencing step is taking the difference of each pixel between the two images. After the differencing step comes the preprocessing step which is a denoising step that removes the obvious background. One of the main benefits from the preprocessing step is that it will be able to improve the performance of the algorithm and the converging speed at the learning stage. Next, the proof of the rotation invariant transform step is introduced in Section 7.3.1. Now the feature extraction is a step that translates a matrix into a vector which shares the same elements. Following feature extraction comes I-RELIEF or Iterative-RELIEF step which is a feature weighting algorithm looking for a weight vector that maximize the margin between two classes and minimize the margin of elements within the same class. Then the classifier is based on the ratio of the distances from the unknown feature of the two classes. Lastly, the post processing works to cluster nearby pixels, remove small detections and the output is the location of every detection. Refer to later part of this section and our previous work [57] for more details.

The rest of this section is organized as below. Section 7.2 briefly describes the data set. In Section 7.3, we present our proposed scheme. Section 7.4 shows the experimental results.

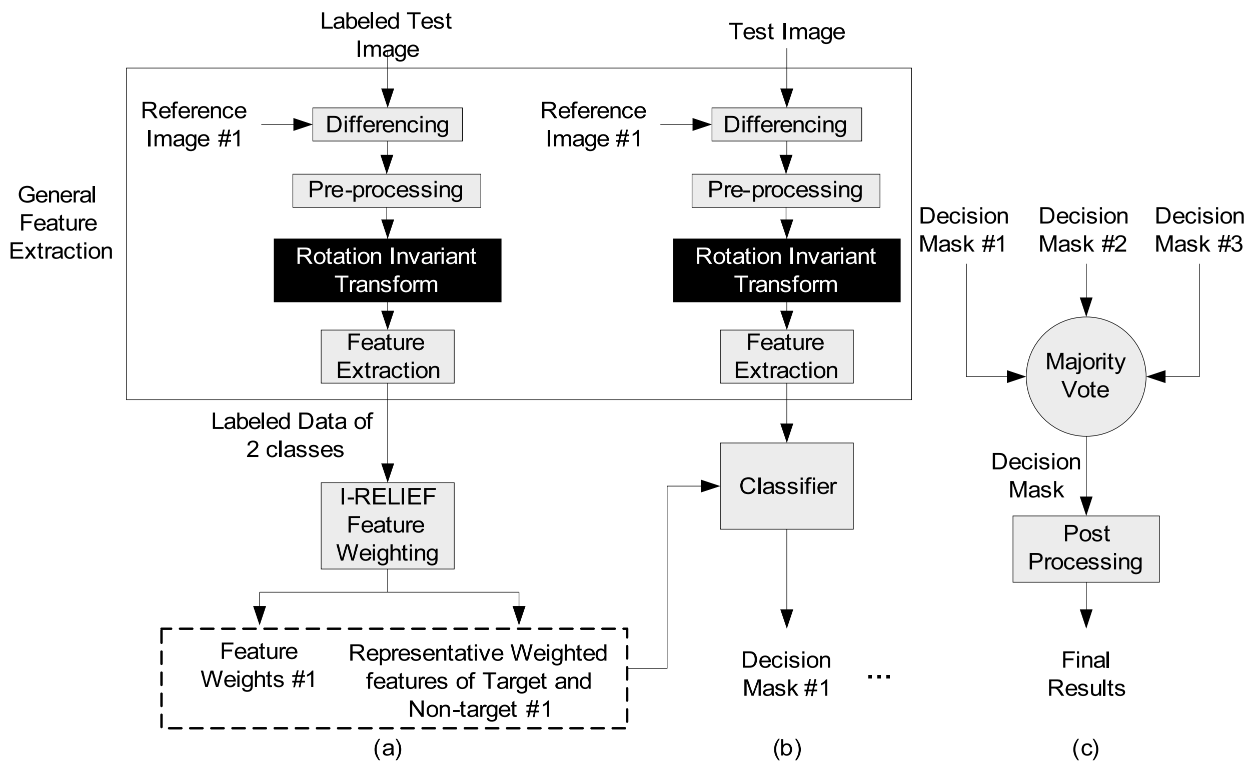


Figure 37: Flow chart of our scheme: (a) learning; (b) testing; (c) diversity combining; black blocks are additional parts in this section.

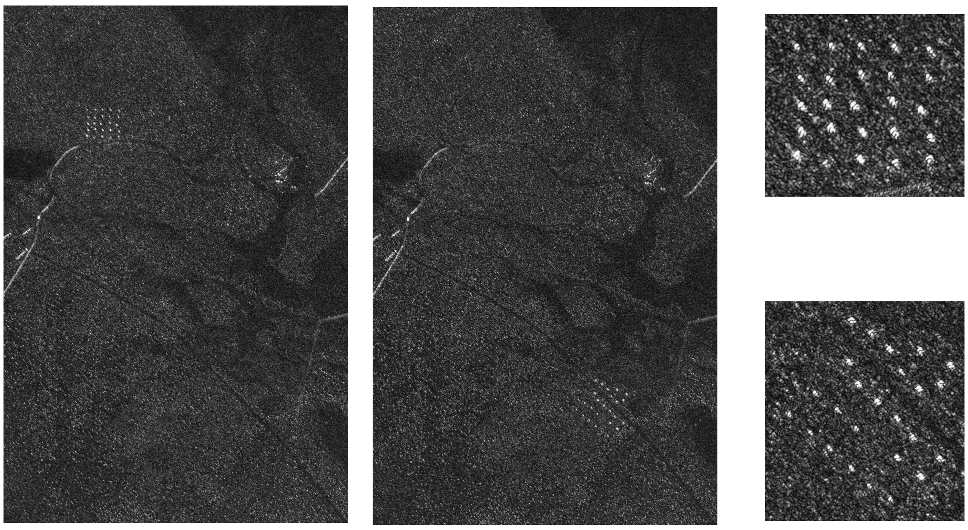


Figure 38: Sample images from dataset. Left: Sigismund deployment, flight heading 225; Middle: Fredrik deployment, flight heading 225; Right: Amplified target regions (25 targets each)

7.2 Data Description

The dataset collected during a flight campaign held in Sweden in summer 2002 was used for the evaluation of the performance of the algorithm. All the images in the dataset were taken by CARABAS-II ultra-wide-band SAR system mounted on a Sabreliner airplane. This system was operated in the frequency range of 20MHz - 90MHz which corresponds to the wavelength of 3.3 meters to 15 meters. The wavelengths of 3.3 meters to 15 meters are comparable to the size of vehicles as targets to be pursued.

In this dataset, all the images are 3000×2000 pixels which are highly accurate intensity matrices that cover the same 3km by 2km ground area; therefore, the resolution of the data is 1 meter per pixel. There are 24 images in the dataset which were taken at 4 different target locations and 6 flights for each location. For each image, the locations and heading of every vehicle, flight heading, incidence angle, and Radio Frequency Interference (RFI) level are given with the dataset. Fig. 38 shows two sample images and the amplified target regions. Please refer to [46] for more details about the dataset.

7.3 Proposed Target Detection Scheme

Fig. 37 shows the flow chart of our proposed scheme. It consists of three parts: learning, testing and majority voting.

Learning and testing share the same procedures of differencing, preprocessing, rotation invariant transform and feature extraction. These four common steps can be noted as general feature extraction shown as Fig. 39. This function module serves to extract a feature vector set $\{\mathbf{x}_i, i = 1, 2, ..., N\}$ from the test image $I_t(x, y)$ and the reference image $I_r(x, y)$. For a given location (x, y) in the image, there would be a corresponding feature vector extracted from a small window centered at (x, y). The small window slides across the image to extract feature vectors from different locations into the output feature vector set. Now to discuss in more detail, the differencing step takes the pixels from the test image and subtracts the pixels from the reference image to remove background noise and clutter. To further suppress the noise in the difference image, both the low pass filter and small threshold are used in this process. First a small threshold is applied to the filtered image in order to remove pixels that are obviously noise. Then the uniform matrix h is used as the convolution kernel of the low pass filter. The calculation of image I_d is shown in equation (42). In the third step, the rotation invariant transform which will be discussed in section 7.3.1 was applied. Finally, the feature extraction step is an element reordering step which puts elements of a matrix into a vector in a specific order.

$$I_d = \begin{cases} h * (I_t - I_r) & h * (I_t - I_r) > th \\ 0 & h * (I_t - I_r) \leq th \end{cases} \tag{42}$$

In the learning stage, locations of all the targets are assumed to be given, so the extracted feature set can be labeled as either “1”, target or “0”, non-target. The labeled feature set is fed to the I-RELIEF feature weighting algorithm to find the best weight vector $\mathbf{w}$ that maximize the margin between two classes of features and minimize the margin within the same class. At the end of this stage, a trained weight vector $\mathbf{w}^*$ and two representative feature vectors from target and non-target classes are stored for future use in the testing stage. Here, the arithmetic average of all the feature vectors within each class is used as a representative feature vector of that class.

For the testing stage, feature vectors are extracted by moving a sliding window across every possible pixel in the image. Then the vectors are fed to a classifier which is explained in section 7.3.3. The output of the classifier is then assigned to the corresponding pixel as a decision of “1” target or “0” non-target. At the end of this stage, a decision mask of a binary valued image is

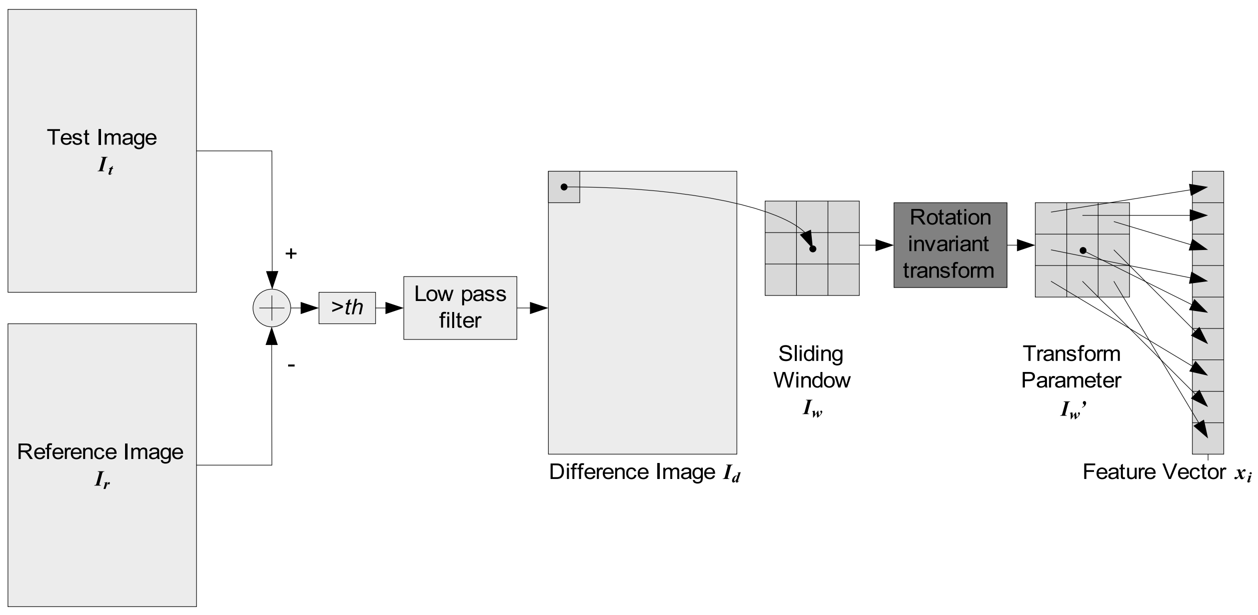


Figure 39: General feature extraction procedure.

exported to the next step.

Finally, the last stage is comprised of majority voting and post processing. Now majority voting independently makes decisions based on several different reference images and chooses the most frequent output as the final decision. Then the purpose of post processing is to connect all the adjacent target pixels into clusters of potential targets, remove those clusters which are too small to be a target, and output the center of each cluster as the location of detections.

There is an algorithm performance evaluation module after all the above steps to compare the location of every detection given by the algorithm with the ground truth locations from the dataset. If an output location is within certain distance range from any ground truth location, a correct detection is claimed, otherwise, a false alarm has occurred. For each ground truth target, only one correct detection can be assigned. Otherwise the detection rate will be inaccurate because the algorithm may claim more detection than what was actually detected.

7.3.1 Rotation Invariant Transform

Local photometric descriptors computed from interest regions such as Scale Invariant Feature Transform (SIFT) [59] have been used in many applications with great success. However, the manipulated local descriptors are not suitable choices in our research because the target is too small

$$\begin{tikzcd} \boldsymbol{I}(\boldsymbol{x},\boldsymbol{y}) \arrow[r, "\text{Radon transform}"] & \boldsymbol{R}(\boldsymbol{b},\theta) \arrow[r, "\substack{\text{1-D Fourier transform on}\theta \\ \text{and take the magnitude}}"] & |\boldsymbol{F}_\theta\{\boldsymbol{R}(\boldsymbol{b},\theta)\}| \end{tikzcd}$$

Figure 40: Rotation invariant transform procedure.

and the image is extremely noisy. In our previous work [57], raw pixel values from a small window are extracted as local features which is the simplest descriptor and indicates good performance of detecting targets.

Raw pixel value descriptor is simple to process and it preserves all the information within the interest region; however, the raw pixel value is a low level local feature which contains a significant amount of redundant information. For example, the same target with different orientations could lead to very different features because the redundant information can easily lead to over fitting or the algorithm diverges in the learning stage if the targets of different orientations were used as training samples.

In our research the detection algorithm was designed to locate targets in SAR images no matter the rotational position of the target. The rotation invariant transform extracts underlying features which is irrelevant to the orientation of the object and use these features to describe the characteristics of the target. Now the extracted rotation invariant feature is an abstraction of the raw data at a higher level which is only related to the target itself but not its orientation. Then, the detection problem is brought into a normalized framework.

Our section uses an algorithm which consists of Radon transform and Fourier transform shown in Fig. 40. Two dimensional Radon transform is the projection of the image intensity along a radial line oriented at a given angle [60]. A straight line AA' in Fig. 41 can be defined parametrically by:

$$\begin{cases} x(t) = bcos\theta + tsin\theta \\ y(t) = bsin\theta - tcos\theta \end{cases} \tag{43}$$

Then, the Radon transform can be written as:

$$R(b,\theta) = \int_{-\infty}^{\infty} I(x(t), y(t))\, dt \tag{44}$$

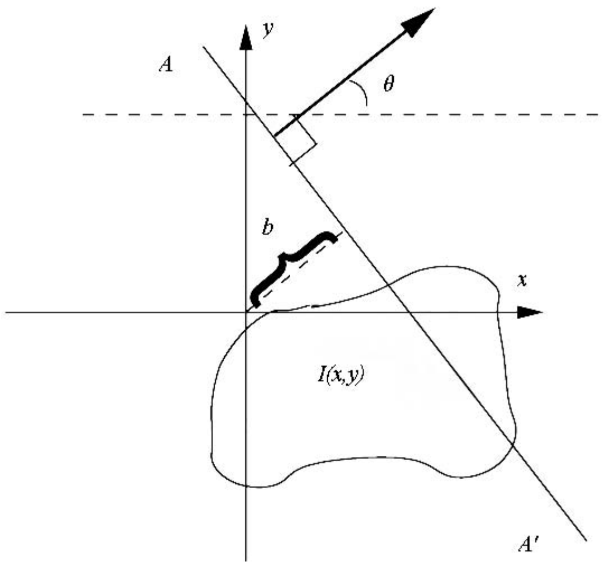


Figure 41: 2-D Radon transform sketch.

or the identical expression:

$$R(b,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x,y)\delta(b - xcos\theta - ysin\theta)dxdy \tag{45}$$

By doing this for different values of b and θ from 0 degree to 180 degree, the original image $I(x,y)$ is transformed into $R(b,\theta)$.

In order to prove that the proposed rotation invariant transform will get the same output for an original image $I(x,y)$ and its rotated version $I'(x,y)$, we will need to prove:

$$|\mathcal{F}_\theta\{\mathcal{R}\{I'(x,y)\}\} = |\mathcal{F}_\theta\{\mathcal{R}\{I(x,y)\}\} \tag{46}$$

where $\mathcal{F}_\theta\{\bullet\}$ is Fourier transform along the direction of θ and $\mathcal{R}\{\bullet\}$ is the Radon transform. According to the geometry knowledge, an image rotated by θ_0 degree counterclockwise becomes:

$$I'(x,y) = I(xcos\theta_0 - ysin\theta_0, xsin\theta_0 + ycos\theta_0) \tag{47}$$

Then, its Radon transform can be written as:

$$\begin{aligned} R'(b,\theta) = \mathcal{R}\{I'(x,y)\} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I'(x,y)\delta(b - xcos\theta - ysin\theta)dxdy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(xcos\theta_0 - ysin\theta_0, xsin\theta_0 + ycos\theta_0)\delta(b - xcos\theta - ysin\theta)dxdy \end{aligned} \tag{48}$$

Define variables m and n as:

$$\begin{cases} m = xcos\theta_0 - ysin\theta_0 \\ n = xsin\theta_0 + ycos\theta_0 \end{cases} \tag{49}$$

Then,

$$\begin{cases} x = mcos\theta_0 + nsin\theta_0 \\ y = -msin\theta_0 + ncos\theta_0 \end{cases} \tag{50}$$

$$dxdy = |J|dmdn, |J| = det \begin{bmatrix} cos\theta_0 & sin\theta_0 \\ -sin\theta_0 & cos\theta_0 \end{bmatrix} = 1 \tag{51}$$

Pluging equation (50) and (51) into (48) will get:

$$R'(b,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(m,n)\delta(b - mcos(\theta+\theta_0) - nsin(\theta+\theta_0))dmdn = R(b,\theta+\theta_0) \tag{52}$$

Take the Fourier transform on both sides and take the magnitude, it becomes:

$$|\mathcal{F}_\theta\{R'(b,\theta)\}| = |\mathcal{F}_\theta\{R(b,\theta+\theta_0)\}| = |e^{j\omega\theta_0}\mathcal{F}_\theta\{R(b,\theta)\}| = |\mathcal{F}_\theta\{R(b,\theta)\}| \tag{53}$$

Then the proof is done. In the same way, it can be proved that for a rotated and translated image $I''(x,y)$:

$$|\mathcal{F}_\theta\{|\mathcal{F}_b\{\mathcal{R}\{I''(x,y)\}\}|\}| = |\mathcal{F}_\theta\{|\mathcal{F}_b\{\mathcal{R}\{I(x,y)\}\}|\}| \tag{54}$$

This is actually a rotation and translation invariant transform. The proof is given but it is not used because the sliding window mechanism which covers every possible translational position is very robust to translational variance and another operation of taking magnitude means more information loss.

Assume $I''(x,y)$ is another version of $I(x,y)$ with θ_0 degrees counterclockwise rotation followed by (x_0,y_0) translation. Then it can be written as:

$$I''(x,y) = I(xcos\theta_0 - ysin\theta_0 + x_0, xsin\theta_0 + ycos\theta_0 + y_0) \tag{55}$$

Its Radon transform is:

$$R''(b,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(xcos\theta_0 - ysin\theta_0 + x_0, xsin\theta_0 + ycos\theta_0 + y_0)\delta(b - xcos\theta - ysin\theta)dxdy \tag{56}$$

Denote:

$$\begin{cases} m = xcos\theta_0 - ysin\theta_0 + x_0 \\ n = xsin\theta_0 + ycos\theta_0 + y_0 \end{cases} \tag{57}$$

Equation (56) turns to be:

$$R''(b,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(m,n)\delta(b + x_0cos(\theta+\theta_0) + y_0sin(\theta+\theta_0) - m\cos(\theta+\theta_0) - n\sin(\theta+\theta_0))dmdn$$
$$= R(b + x_0cos(\theta+\theta_0) + y_0sin(\theta+\theta_0), \theta+\theta_0) \quad (58)$$

Take the Fourier transform along the direction of b on both sides of equation (58):

$$\mathcal{F}_b\{R''(b,\theta)\} = e^{x_0cos(\theta+\theta_0)+y_0sin(\theta+\theta_0)}\mathcal{F}_b\{R(b,\theta+\theta_0)\} \quad (59)$$

And:

$$|\mathcal{F}_b\{R''(b,\theta)\}| = |\mathcal{F}_b\{R(b,\theta+\theta_0)\}| \quad (60)$$

Then, further proof of equation (54) is to take the Fourier transform along θ direction on both sides of equation (60) and take the magnitude.

In this research we used the rotation invariant features instead of the raw pixel values used in our previous work [57] because the rotation invariant algorithm can detect targets of different orientations whereas raw pixel values algorithm is sensitive to change of orientation. Having the capability of being insensitive to different orientations will alleviate the potential of over fitting problem with raw pixel value features. However, the proposed rotation invariant transform needs to take the magnitude of the Fourier transform coefficients which loses all the phase information, so the new feature is less accurate than the raw feature. When using the rotation invariant features the accuracy is traded for generality.

7.3.2 I-RELIEF

Feature weighting transforms the original feature vector **x** into a new feature vector **x**′ by assigning each feature a positive weight $w^{(i)}$. The feature and weight vectors can be defined as:

$$\mathbf{x} = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(I)} \end{bmatrix} \mathbf{w} = \begin{bmatrix} w^{(1)} \\ w^{(2)} \\ \vdots \\ w^{(I)} \end{bmatrix} \mathbf{x}' = \begin{bmatrix} x'^{(1)} \\ x'^{(2)} \\ \vdots \\ x'^{(I)} \end{bmatrix} \quad (61)$$

where $x'^{(i)} = x^{(i)}w^{(i)},\ i = 1, 2, \ldots, I$, I is the data dimensionality.

I-RELIEF or Iterative-RELIEF is an improved version of RELIEF which is a feature weighting algorithm for increasing the discrimination between classes. The key idea of RELIEF is to solve a convex optimization problem with a margin-based objective function:

$$\max_{\mathbf{w}} \sum_{n=1}^{N} (\sum_{i=1}^{I} w^{(i)} |x_n^{(i)} - NM^{(i)}(x_n)| - \tag{62}$$

$$\sum_{i=1}^{I} w^{(i)} |x_n^{(i)} - NH^{(i)}(x_n)|)$$

$$\text{s.t. } \|\mathbf{w}\|_2^2 = 1, \mathbf{w} \geq 0 \tag{63}$$

NM means the nearest miss of **x** and NH means the nearest hit of **x**. Two problems with RELIEF are that NM and NH are defined in the original feature space and the outliers can dramatically influence the margin calculation. To solve the two problems, I-RELIEF calculates the margin based on the probabilities of NM, NH, and outliers estimated in the weighted feature space and updates the weights iteratively. Refer to [55] for details.

7.3.3 Classifier

The structure of our classifier is shown in Fig. 42 which is the same as our previous work [57]. Also, our classifier shares the similar philosophy with Neyman-Pearson detector. Now looking at Fig. 42, D_1 and D_0 are all Euclidean distances and different values of threshold causes a trade off between detection rate and false alarm rate.

7.4 Experimental Results

To evaluate the performance of our algorithm, we set up experiments based on the 24 public SAR images included in the CARABAS-II radar data set. These images are categorized as mission 2, 3, 4 and 5 according to the 4 different vehicle locations. For each location, 6 images known as pass 1 through 6, were taken in different operating conditions such as flight heading, incidence angle, and radio frequency interference level. Once we receive a test image we always chose the reference image from the different locations but under the same operating conditions; therefore, for any test image, there could be 3 independent reference images for three independent training processes.

The image of mission 3 pass 5 was always used as the test image in the learning stage from which target and non-target training sets were extracted. Now the I-Relief feature weighting algorithm will generate an optimized weight vector $\mathbf{w}^*$ from the two training sets. We put the $\mathbf{w}^*$ weight vector together with the averages of the two sets of training samples into our classifier. The above procedures were repeated for each reference image to finish the learning stage. Finally,

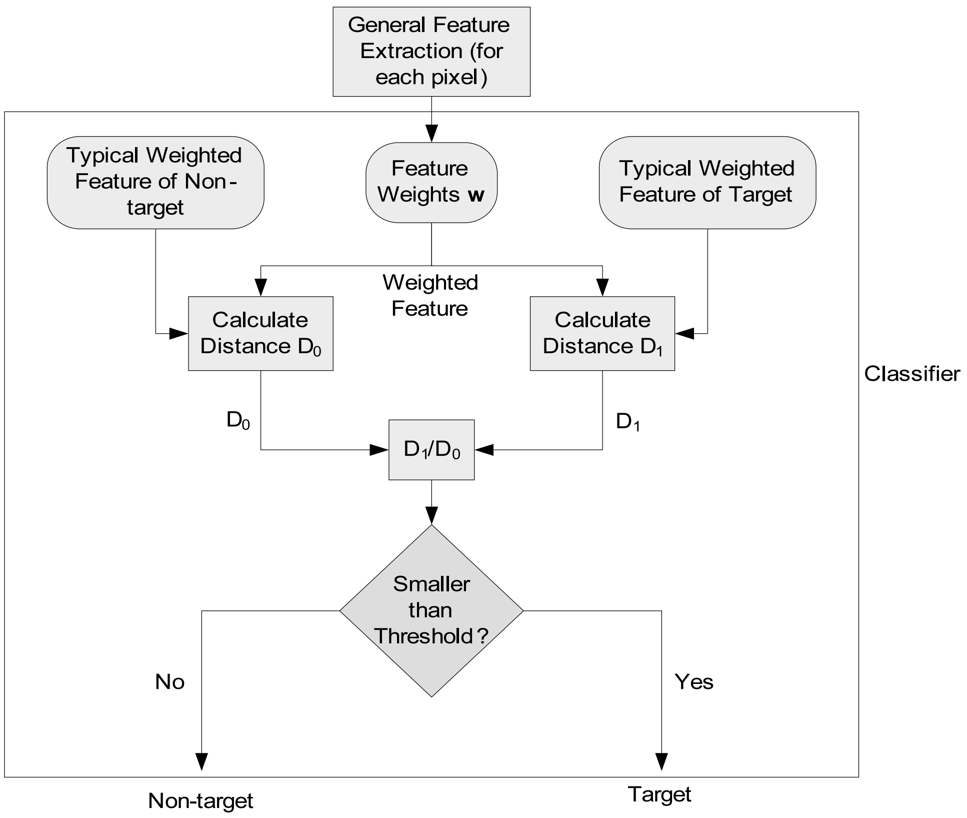


Figure 42: The diagram of classification.

we obtained three classifiers for the three reference images with the same structure but different parameters.

In the testing stage for each test and reference image pair, the classifier trained in the learning stage will make a decision for each pixel and output a decision mask. Since there are three independent reference images there will have three independent decision masks which by majority voting will merge into one final decision mask. Then post processing of clustering will apply to the mask and the centroid of a large enough cluster will mark the position of the detection. When evaluating the performance of our algorithm, we compare the position of the detections with the locations of real targets and if the distance is less than 10 pixels (i.e. 10 meters), we claim that one detection has been made. For each real target location, only one detection can be claimed and the rest of our detections are false alarms.

The above experiment setup is the same as our previous work [57], except an additional rotation invariant module was added to the experiment which is implemented with the help of Radon Transform and Fast Fourier Transform (FFT). Actually Radon Transform requires discrete values of θ between $0°$ and $180°$ as inputs. In our experiment, θ is chosen to be $0°, 15°, 30°, 45°, ..., 180°$. When the Radon Transform is applied to the sliding window of 19×19 with the given θ values, the feature vector increases its dimension from 361 to 377. Since the output of the Radon transform is a 29×13 instead of 19×19 matrix and FFT does not change its dimension.

Table 3 shows the parameters used in our experiment. Table 4 shows the experimental results from our algorithm compared with the benchmark algorithm. However, image of mission 3 pass 5 was not used for testing because it served as a training set. In conclusion, our experimental results show that our algorithm produces a lower false alarm rate and higher detection rate than the benchmark algorithm. For example, our algorithm produced 25 false alarms and missed one detection and the benchmark algorithm produced 86 false alarms and missed 13 detections.

8 Depth Based Image Registration

The objective of Task 6 is to devise an image registration algorithm that is capable of mitigating the parallax problem. We develop a depth based image registration scheme to mitigate the parallax problem.

Current image registration algorithms suffer from the parallax problem due to the assumption that all object points in 3D space are on the same plane. However, this assumption is not valid for aerial images captured by cameras mounted on low-flying airplanes. For this scenario, the high rise buildings will cause significant parallax problems, i.e., the close-by high rise buildings

Processing step	Parameter	Value
Preprocessing	Averaging kernel size	5×5 pixels
	Denoising threshold	0.25
Rotation invariant transform	Discrete θ values	$0°, 15°, 30°, 45°, ..., 180°$
Feature extraction	Sliding window size	19×19 pixels
I-RELIEF feature weighting	Maximum number of iterations	500
	Distance metric	‘Euclidean’
	Kernel function	$f(d) = exp(-d/\sigma)$
	Kernel width σ	25
Classification	Threshold λ on DR	3.0
Post processing	Minimum number of connected pixels as a target	35 pixels
Evaluation	Distance threshold	10 pixel

Table 3: Parameters used to test performance

move fast than the stationary objects on the ground in the video sequence. Hence, if we use a conventional image registration scheme (which assumes that all object points in 3D space are on the same plane), in the registered video sequence, we could observe artificial motion between the high rise buildings and the stationary objects on the ground. The only way to address this problem is to use 3D information.

Hence, we propose a depth based image registration scheme. Our key idea is to use depth information. To achieve this, we need to extract depth from 2D video sequence. Our technique is called ”structure from motion”. We implement the depth based image registration scheme. Experimental results show that our scheme achieves much better performance than the conventional image registration scheme (which assumes that all object points in 3D space are on the same plane), if there is significant parallax. Our results show that depth information can really help mitigate the parallax problem.

The results of Task 6 were published in Ref. [61]. Next, we present the technical details.

8.1 Introduction

Image registration is a fundamental task in image processing and computer vision which matches two or more images taken at different times and different viewpoints, by geometrically aligning reference and sensed images. There has been a broad range of techniques developed over the years in literature. A comprehensive survey of image registration methods was published in 1992 by Brown [62], including many classic methods still in use. Due to the rapid development of image

Target image		Correct detections		False alarms	
Mission	Pass	Benchmark Algorithm	Our Scheme	Benchmark Algorithm	Our Scheme
2	1	25	25	2	0
3	1	22	25	1	4
4	1	25	25	2	0
5	1	23	25	4	2
2	2	25	25	2	1
3	2	25	25	4	1
4	2	25	25	3	3
5	2	25	25	4	2
2	3	25	25	3	2
3	3	23	25	4	1
4	3	25	25	0	1
5	3	24	25	2	0
2	4	24	25	3	0
3	4	25	25	2	0
4	4	25	25	4	1
5	4	25	24	4	1
2	5	25	25	3	1
3	5	(Used for training)			
4	5	25	25	2	2
5	5	23	25	29	0
2	6	25	25	1	1
3	6	25	25	3	1
4	6	25	25	1	1
5	6	23	25	3	0
Total		562	574	86	25

Table 4: Comparing of results from benchmark algorithm and that from proposed scheme. Image of mission 3 pass 5 should not be counted in testing because it is used for training.

acquisition devices, more image registration techniques emerged afterwards and were covered in another survey [63] published in 2003. Different applications due to distinct image acquisition require different image registration techniques. In general, manners of the image acquisition can be divided into three main groups:

- *Different viewpoints (multiview analysis)*. Images of the same scene are acquired from different viewpoints. The aim is to gain a larger 2D view or a 3D representation of the scanned scene.
- *Different times*. Images of the same scene are acquired at different times, often on regular basis, and possibly under different conditions. The aim is to find and evaluate changes in the scene which appeared between the consecutive image acquisitions.
- *Different sensors*. Images of the same scene are acquired by different sensors. The aim is to integrate the information obtained from different source streams to gain more complex and detailed scene representation.

Due to the diversity of images to be registered and various types of degradations, it is impossible to design a universal method applicable to all registration tasks. Every method should take into account not only the assumed type of geometric deformation between the images but also the radiometric deformations and noise corruption, required registration accuracy and application-dependent data characteristics. Nevertheless, the majority of the registration methods consists of the following four steps: feature detection, feature matching, transform model estimation, image resampling and transformation.

A widely used feature detection method is corner detection. Kitchen and Rosenfeld [64] proposed to exploit the second-order partial derivatives of the image function for corner detection. Dreschler and Nagel [65] searched for the local extrema of the Gaussian curvature. However, corner detectors based on the second-order derivatives of the image function are sensitive to noise. Thus Forstner [66] developed a more robust, although time consuming, corner detector, which is based on the first-order derivatives only. The reputable Harris detector [67] also uses first-order derivatives for corner detection.

Feature matching includes area-based matching and feature-based matching. Classical area-based method is cross-correlation (CC) [68] exploit for matching image intensities directly. For feature-based matching, Goshtasby [69] described the registration based on the graph matching algorithm. Clustering technique, presented by Stockman et al. [70], tries to match points connected by abstract edges or line segments.

After the feature correspondence has been established the mapping function is constructed. The mapping function should transform the sensed image to overlay it over the reference image.

Finally interpolation methods such as nearest neighbor function, bilinear, and bicubic functions are applied to the output of the registered images.

The prevailing image registration methods, such as the algorithm of Davis and Keck [71, 72], assume all the feature points are coplanar and build a homography transform matrix to do registration. The advantage is that they have low computational cost and can handle planar scenes conveniently; however, with the assumption that the scenes are approximately planar, they are inappropriate in the registration applications when the images have large depth variation due to the high-rise objects, known as the parallax problem. Parallax is an apparent displacement of difference of orientation of an object viewed along two different lines of sight, and is measured by the angle or semi-angle of inclination between those two lines. Nearby objects have a larger parallax than further objects when observed from different positions. Therefore, as the viewpoint moves side to side, the objects in the distance appear to move slower than the objects close to camera.

In this section, we propose a depth based image registration algorithm by leveraging the depth information. Our method can mitigate the parallax problem caused by high-rise scenes in the images by building accurate transform function between corresponding feature points in multiple images. Given an image sequence, we first select a number of feature points and then match the features in all images. Then we estimate the depth of each feature point from feature correspondences. With the depth information, we can project the image in 3D instead of using a homography transform. Further more, fast and robust image registration algorithm can be achieved by combining the traditional image registration algorithms and depth based image registration method proposed in this section. The idea is that we first compute the 3D structure of a sparse feature points set and then divide the scene geometrically into several approximately planar regions. For each region, we can perform a depth based image registration. Accordingly, robust image registration is achieved.

The remainder of this section is organized as follows. Section 12.3 presents the 3D reconstruction algorithm on which our experiments are based. In Section 12.4, we describe our depth-based image registration algorithm from theoretical aspects to explain why it can mitigate the parallax problem. Section 12.5 compares our algorithm with the algorithm of Davis and Keck [71] on the same video sequence.

8.2 3D reconstruction from video sequences

Here, we simply introduce the 3D reconstruction algorithm proposed in Ma et. al's book [73] on which our experiments are based. When developing a stereo vision algorithm for registration, the requirements for accuracy vary from those of standard stereo algorithms used for 3D reconstruction. For example, a multi-pixel disparity error in an area of low texture, such as a white wall, will result in significantly less intensity error in the registered image than the same disparity error in a highly textured area. In particular, edges and straight lines in the scene need to be rendered correctly.

The 3D reconstruction algorithm is implemented using the following steps [73]. First, geometric features are detected automatically in each individual images. Secondly, feature correspondence is established across all the images. Then the camera motion is retrieved and the camera is calibrated. Finally the Euclidean structure of the scene is recovered.

8.2.1 Feature selection

The first step in 3D reconstruction is to select candidate features in all images for tracking across different views. Ma et al. [73] use point feature in reconstruction which is measured by Harris' criterion [],

$$C(\mathbf{x}) = \det(G) + k \times \text{trace}^2(G), \tag{64}$$

where $\mathbf{x} = [x, y]^T$ is a candidate feature, $C(\mathbf{x})$ is the quality of the feature, k is a pre-chosen constant parameter and G is a 2×2 matrix that depends on **x**, given by

$$G = \begin{bmatrix} \sum_{W(\mathbf{x})} I_x^2 & \sum_{W(\mathbf{x})} I_x I_y \\ \sum_{W(\mathbf{x})} I_x I_y & \sum_{W(\mathbf{x})} I_y^2 \end{bmatrix} \tag{65}$$

where $W(\mathbf{x})$ is a rectangular window centered at **x** and I_x and I_y are the gradients along the x and y directions which can be obtained by convolving the image I with the derivatives of a pair of Gaussian filters. The size of the window can be decided by the designer, for example 7×7. If $C(\mathbf{x})$ exceeds a certain threshold, then the point **x** is selected as a candidate point feature.

8.2.2 Feature correspondence

Once the candidate point features are selected, the next step is to match them across all the images. In this subsection, we use a simple feature tracking algorithm based on a translational model.

We use the sum of squared differences (SSD) [74] as the measurement of the similarity of two point features. Then the correspondence problem becomes looking for the displacement **d** that satisfies the following optimization problem:

$$\min_{\mathbf{d}} \doteq \sum_{\mathbf{x} \in W(\mathbf{x})} [I_2(\mathbf{x} + \mathbf{d}) - I_1(\mathbf{x})]^2 \tag{66}$$

where $\mathbf{d}$ is the displacement of a point feature of coordinates $\mathbf{x}$ between two consecutive frames I_1 and I_2. Lucas and Kanade also give the close form solution of 169

$$\mathbf{d} = -G^{-1}\mathbf{b} \tag{67}$$

where

$$\mathbf{b} \doteq \begin{bmatrix} \sum_{W(\mathbf{x})} I_x I_t \\ \sum_{W(\mathbf{x})} I_y I_t \end{bmatrix} \tag{68}$$

G is the same matrix we used to compute the quality of the candidate point feature in Eq. 167, and $I_t \doteq I_2 - I_1$.

8.2.3 Estimation of camera motion parameters

In this subsection, we recover the projective structure of the scene from the established feature correspondence. We will follow the notation used in Ma et al.'s book [73]. For the detail of the proof of this algorithm, please refer to the reference.

The reconstruction algorithm is based on a perspective projection model with a pinhole camera. Suppose we have a generic point $p \in \mathbb{E}^3$ with coordinates $\mathbf{X} = [X, Y, Z, 1]^T$ relative to a world coordinate frame. Given two frames of one scene which is related by a motion $g = (R, T)$, the two image projection point $\mathbf{x}_1$ and $\mathbf{x}_2$ are related as follows:

$$\lambda_1 \mathbf{x}_1' = \Pi_1 \mathbf{X}_p, \quad \lambda_2 \mathbf{x}_2' = \Pi_2 \mathbf{X}_p, \tag{69}$$

where $\mathbf{x}' = [x, y, 1]^T$ is measured in pixels, λ_1 and λ_2 are the depth scale of $\mathbf{x}_1$ and $\mathbf{x}_2$, $\Pi_1 = [K, 0]$ and $\Pi_2 = [KR, KT]$ are the camera projection matrices and K is the camera calibration matrix. In order to estimate λ_1, λ_2, Π_1 and Π_2, we need to introduce the epipolar constraint. From Eq. 172, we have

$$\mathbf{x}_2'^T K^{-T} \hat{T} R K^{-1} \mathbf{x}_1' = 0. \tag{70}$$

The fundamental matrix is defined as:

$$F \doteq K^{-T} \hat{T} R K^{-1}. \tag{71}$$

Table 5: Eight-point algorithm

Given a set of initial point feature correspondences expressed in pixel coordinates $(\mathbf{x'}_1^j, \mathbf{x'}_2^j)$ for $j = 1, 2, ..., n$:

• **A first approximation of the fundamental matrix:** Construct the matrix $\chi \in \mathbb{R}^{n\times 9}$ from the transformed correspondences $\tilde{\mathbf{x}}_1^j \doteq [\tilde{x}_1^j, \tilde{y}_1^j, 1]^T$ and $\tilde{\mathbf{x}}_2^j \doteq [\tilde{x}_2^j, \tilde{y}_2^j, 1]^T$, where the jth row of χ is given by $[\tilde{x}_1^j\tilde{x}_2^j, \tilde{x}_1^j\tilde{y}_2^j, \tilde{x}_1^j, \tilde{y}_1^j\tilde{x}_2^j, \tilde{y}_1^j\tilde{y}_2^j, \tilde{y}_1^j, \tilde{x}_2^j, \tilde{y}_2^j, 1]^T \quad \in \mathbb{R}^9$. Find the vector $F^s \in \mathbb{R}^9$ of unit length such that $||\chi F^s||$ is minimized as follows: Compute the singular value decomposition (SVD) of $\chi = U\Sigma V^T$ and define F^s to be the ninth column of V. Unstack the nine elements of F^s into a square 3×3 matrix $\tilde{F}$.

• **Imposing the rank-2 constraint:** Compute the SVD of the matrix F recovered from data to be $\tilde{F} = U_F diag\{\sigma_1, \sigma_2, \sigma_3\}V_F^T$. Impose the rank-2 constraint by letting $\sigma_3 = 0$ and reset the fundamental matrix to be $F = U_F diag\{\sigma_1, \sigma_2, 0\}V_F^T$.

With the above model, we could estimate the fundamental matrix F via the Eight-point algorithm [73]. Then we could decompose the fundamental matrix to recover the projection matrices Π_1 and Π_2 and the 3D structure. We only give the solution here by canonical decomposition:

$$\Pi_1 p = [I, 0], \Pi_2 p = [(\widehat{T'})^T F, T'], \lambda_1 \mathbf{x}'_1 = \mathbf{X}_p, \lambda_2 \mathbf{x}'_2 = (\widehat{T'})^T F \mathbf{X}_p + T'. \tag{72}$$

8.2.4 Depth estimation

The Euclidean structure $\mathbf{X}_e$ is related to the projective reconstruction $\mathbf{X}_p$ by a linear transform $H \in \mathbb{R}^{4\times 4}$,

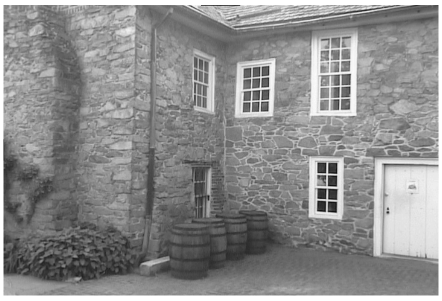
$$\Pi_{ip} \sim \Pi_{ie} H^{-1}, \mathbf{X}_p \sim H\mathbf{X}_e, i = 1, 2, ..., m, \tag{73}$$

where $\sim$ means equality up to a scale factor and

$$H = \begin{bmatrix} K_1 & 0 \\ -\nu^T K_1 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}. \tag{74}$$

With the assumption that K is constant, we could estimate the unknowns K and ν with a gradient decent optimization algorithm. In order to obtain a unique solution, we also assume that the scene is generic and the camera motion is rich enough.

Fig. 75 shows the first frame and the 88th frame of the test video sequence ‘oldhousing’. In our experiment, we will register all the frames in the video sequence to the first frame. Fig. 76 show the selected feature points on the first frame which are used for camera pose estimation. Fig. 77 show the estimated depth map of the selected feature points and the camera pose.


(a) The 1st frame in the 'oldhousing' video sequence

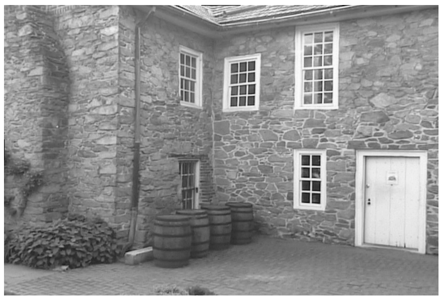

(b) The 88th frame in the 'oldhousing' video sequence

Figure 43: Original frames used for image registration

8.3 Image registration with depth information

Once we obtain the 3D structure of the feature points, the motion, and calibration of the camera, we can start to register the rest of the pixels in the images with the estimated depth information. The traditional image registration algorithms, such as the algorithm of Davis and Keck [71,72], try to register the two images by computing the homography matrix H between corresponding feature points. The limit of this algorithm is that they assume all the points in the physical world are coplanar or approximately coplanar, which is not true with high-rise scenes. In order to mitigate this problem, we propose a novel algorithm which first segment the image geometrically and then perform the registration to each region with depth estimation.

8.3.1 Geometrical segmentation

In order to perform the geometrical segmentation, the most intuitive method is to obtain the dense surface model of the scene and then segment the surface into several regions based on the depth of the points. However, we need to know the correspondence for almost all the pixels to compute the dense surface model, which means we need to know all the pixel correspondence before the registration. In order to avoid this dilemma, we will not use the traditional 3D reconstruction algorithm to estimate the dense surface model. Instead, we directly segment the scene into several regions by clustering the sparse 3D points set that we obtained in Section 12.3. With the assumption

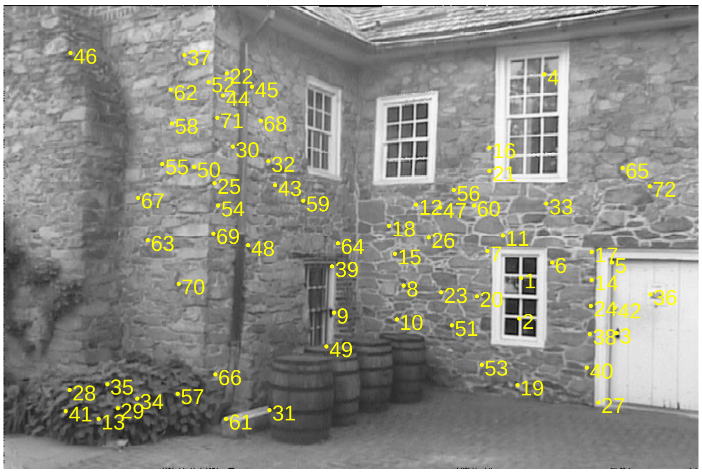


Figure 44: The feature points selected for depth estimation on the 1st frame.

that each segment region of the scene is approximately coplanar in the physical world, we could easily estimate the plane model and project the 3D plane onto the image frames. Comparing the assumption that the whole scene is coplanar in the physical world used in the traditional image registration algorithms, this assumption is valid in most circumstances.

There are a lot of algorithms for data clustering. The most famous hard-clustering algorithm is k-means [75]. The k-means algorithm assigns each data point to the cluster whose centroid is nearest. Here, we use the distance to a 3D plane in the physical world as the measurement. For each cluster, we could choose the plane that has the smallest sum of distance of all the data points in the cluster.

8.3.2 Depth estimation

Here, we only consider two images. Suppose for the first image, we have the 3D point set $\mathbf{X}_e^j, j = 1, 2, ..., n$ which could be divided into three clusters,$\mathbf{X}_{e1}$, $\mathbf{X}_{e2}$, $\mathbf{X}_{e3}$. For each cluster, there are at least three non-collinear points. Then we could have the plane model for this cluster. Let's take the example of $\mathbf{X}_{e1}$, suppose there are m points in the cluster and we have the plane model as follows:

$$\mathbf{A} \cdot p = 1. \tag{75}$$

where $\mathbf{A} = [\mathbf{X}_{e1}^i], i = 1, ..., m$ and $p = [a, b, c]^T$ is the plane parameter.

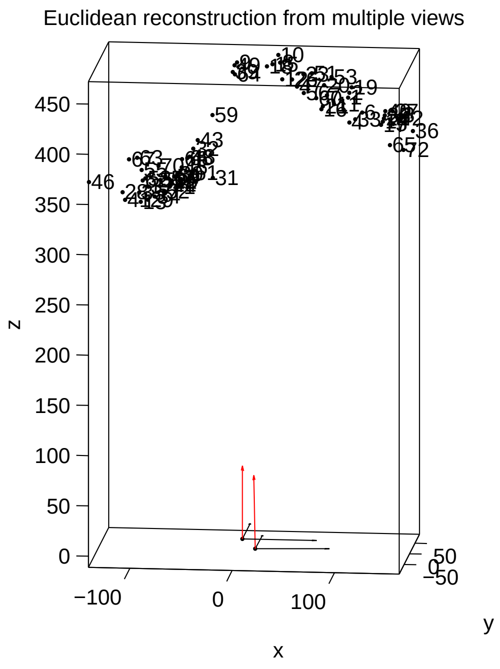


Figure 45: The estimated depth map and camera pose for the selected feature points of the 1st and 88th frames.

Given an arbitrary point $\mathbf{x}^i = [x^i, y^i]^T$ measured in pixels in the first cluster, we could estimate it's depth scale λ^i by solving the following equation.

$$\lambda^i \mathbf{x}'^i = H_1^{-1} \Pi_1 \mathbf{X}_e^i. \tag{76}$$

where $\mathbf{x}'^i = [x^i, y^i, 1]^T$, H_1^{-1} and Π_1 are estimated in Section 12.3. In Eq. 196, only λ^i is unknown and with the constraint on $\mathbf{X}_e^i$ with Eq. 195, we could easily get the value of λ^i.

Then, with $\Pi_1 = [I, 0]$, we could have $X_p^i = [\lambda_1^i x^i, \lambda_1^i y^i, \lambda_1^i, 1]$. from Eq. 172, we can get the relation between two image projection point $\mathbf{x}_1^i$ and $\mathbf{x}_2^i$ as follows:

$$\widehat{\mathbf{x_2^i}'} = \Pi_2 \mathbf{X}_p^i. \tag{77}$$

where $\widehat{\mathbf{x_2^i}'} = [\lambda_2^i x_2^i, \lambda_2^i y_2^i, \lambda_2^i]$. We could then get the position of the corresponding point $\mathbf{x}_2^i = [x_2^i, y_2^i]$ in the second image.

8.4 Experimental Results

The data includes a sequence of 88 images captured from one camera. We first select 72 feature points in the first image and then find the corresponding feature points in the rest of the images.

The depth estimates of these points are calculated by the algorithm introduced in Section 12.3.

In our experiment, we regard the first image's local coordinate system as world coordinate system so the first image can be viewed as a reference image. Then the rest of the images are registered to the reference image. We also applied the algorithm of Davis and Keck [71] to accomplish the same task for comparison purpose.

Fig. 75 is the 1st frame and the 88th frame in the test image sequence. Fig. 79 is the registration result using our algorithm and Fig. 80 is the output of the algorithm of Davis and Keck [71]. Fig. 81 shows the difference image between the registered image and the first image using our algorithm and Fig. 82 shows the difference image from the algorithm of Davis and Keck [71]. We can see that our result can mitigate the parallax problem since the roof and wall corners are registered correctly; on the contrary, the registered image by the algorithm of Davis and Keck [71] has a lot of artifacts caused by the parallax problem. We also show some registration results using our algorithm in Fig. 83$\sim$ Fig. 84.

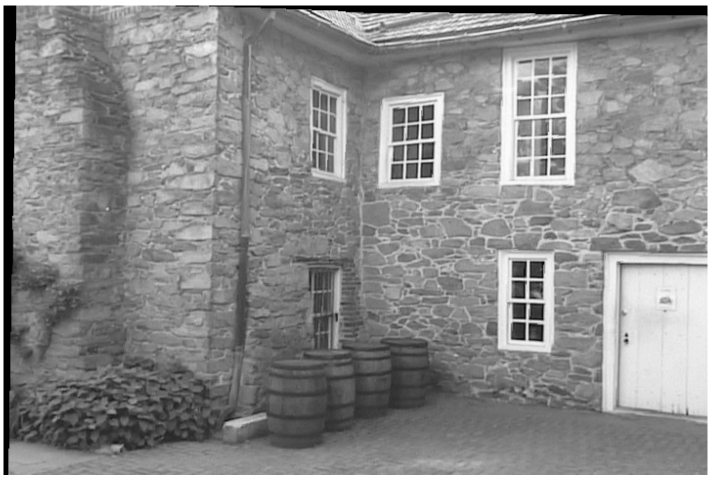


Figure 46: Our algorithm test result, in which the 88th frame is registered to the 1st frame.

In order to further compare our algorithm to the algorithm of Davis and Keck, we compute the root of mean squared errors (RMSE) of the registration results from both algorithms. Fig. 85 shows that the registration error of our algorithm is less than 50% than that of the algorithm of Davis and Keck.

The result shows that our image registration algorithm can mitigate the parallax problem be-

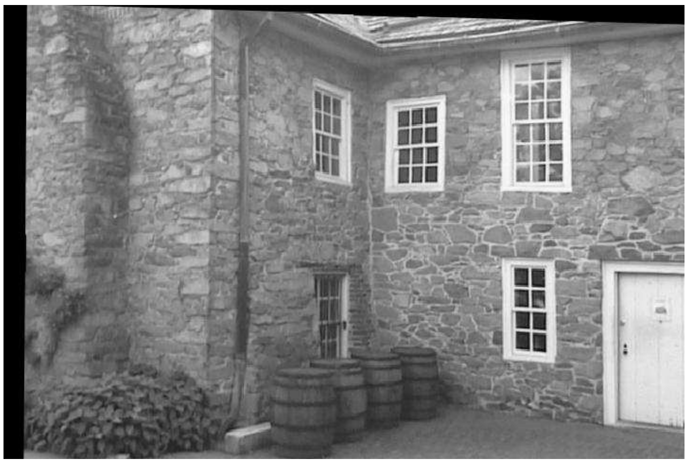


Figure 47: The test result under the algorithm of Davis and Keck, in which the 88th frame is registered to the 1st frame.

cause most of the scene is registered without vibration, as opposed to registration results under the algorithm of Davis and Keck in which the high-rise scene in the sensed images significantly moved after registration to the reference images. The reason is that the algorithm of Davis and Keck assumes all the points in the images are coplanar. While this assumption is satisfied when the distance between the camera and the interested scene is so large that the small depth variation can be neglected, it fails in the case of high-rise scene. Therefore, depth information should be used to accomplish the registration for this specific high-rise region of the images.

Finally, we would like to point out that the algorithm of Davis and Keck [71] assumes a planar registration. Their scheme was designed for use with high-altitude aerial imagery where planar transformations are fairly good approximations. Furthermore, their scheme uses RANSAC to remove poor matching points during the computation. This can help to deal with some depth discontinuities that may be present in the high-altitude aerial images. In our experiments, the test images contain salient 3D scenes; these images are out of the domain for the algorithm of Davis and Keck. This is the reason why the algorithm of Davis and Keck does not perform well.

9 Wavelet-Based Image Registration

The objective of Task 7 is to develop a wavelet based image registration scheme that can achieve low Root Mean Squared Error (RMSE) in registration.

The goal is to create an algorithm that reduces the Root Mean Squared Error (RMSE) around 4 and increases the Peak to Signal Noise Ratio (PSNR) above 34 by using the wavelet coefficients to extract feature points, do feature point correspondence, and register images. Since this research

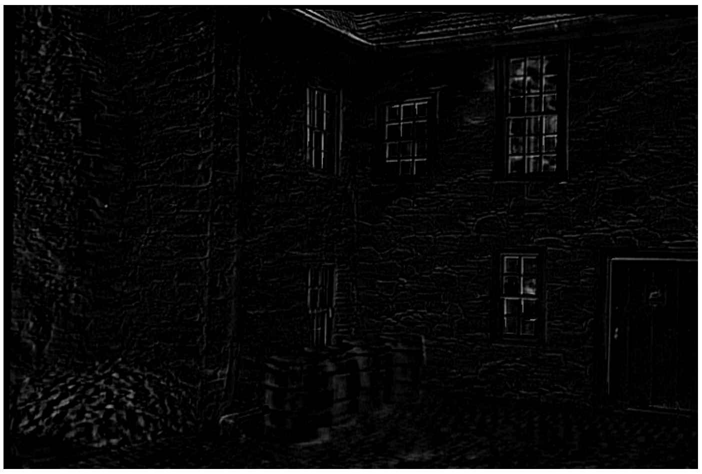


Figure 48: The difference image between the registered 88th image (using our algorithm) and the 1st image.

deals with wavelet coefficients, two algorithms were developed; one algorithm uses the approximate coefficients, while the other uses the detailed coefficients. The preliminary results show that the algorithm is able to achieve a PSNR of approximately 33 and RMSE of approximately 5.

The results of Task 7 were published in Ref. [76].

Next, we present the technical details.

9.1 Introduction

To introduce the subject of image registration, it is important to understand that it is the fundamental enabling technology in computer vision that aligns two or more images together taken at different times (multitemporal analysis), viewpoints (multiview analysis), and/or sensors (multimodal analysis) [77]. Developing an accurate image registration algorithm will significantly improve the techniques for computer vision problems such as tracking, fusion, change detection, autonomous navigation. There has been a significant amount of research that has been conducted in developing image registration algorithms and some of the algorithms have been complied in a survey by Brown in 1992 [78] and Zitova and Flusser in 2003 [77]. However, image registration has not been solved yet because the algorithms are not robust since the algorithm is only able to register particular types of images due to the parameter settings and not able to register other images [79]. Another prob-

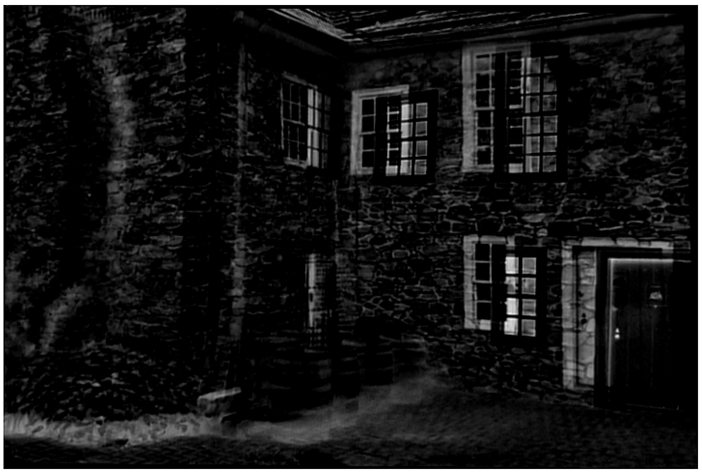


Figure 49: The difference image between the registered 88th image (using the algorithm of Davis and Keck) and the 1st image.

lem within image registration is the parallax problem caused by the high rise buildings because the buildings appear to be swaying; therefore, causing other computer vision algorithms to have inadequate performances. Also registration algorithm needs to perform faster than real time, so other algorithms such as a tracking can track an object in real time which will be vital in defense applications. However, image registration does not have algorithm that is able to perform at such a speed due to high computational complexity.

Image registration typically consists of the following steps:

1. **Preprocessing**: modifies both the sensed (input) and reference (base) image in order to improve the performance of the feature selection and feature correspondence of image registration because some images may be blurry or have significant amount of noise which will dramatically affect the outcome of the algorithm [79]. Some techniques alleviating the noise (Image Smoothing) are median filters, mean filters, gaussian filters, etc [79]. Also, the techniques for deblurring (Image Sharpening) are the Laplacian, high boost filtering, gradient, etc [80].

2. **Feature Extraction**: selects the key features such as corners, lines, edges, contours, templates, regions, etc. which will be used to do feature correspondence [79]. Some examples of feature selection are Harris corner detector, gradient, Hough transform, etc [79].

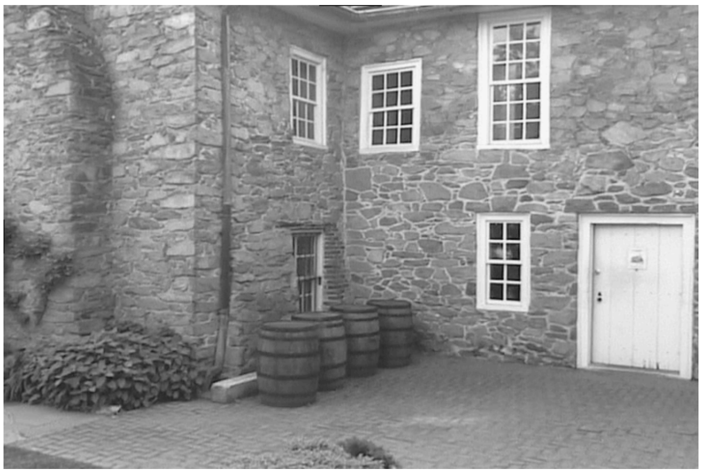


Figure 50: The 37th frame in the ‘oldhousing’ video sequence.

3. **Feature Correspondence**: matches the key features selected in the reference image and the sensed image to see which points in the reference image matches with the points in the sensed image [79]. Cross correlation, mutual information, template matching, Chamfer, etc. are a few examples of feature correspondence [79].

4. **Transformation Function** aligns the sensed image to the reference image by the mapping function [79]. A few example of transformation functions are affine, projective, piecewise linear, thin-plate spline, etc [79].

5. **Resampling** takes the coordinate points location of the discrete points and transforms them into a new coordinate system because the sensed image is an uniformly spaced sample of a continuous image [79, 81]. Some examples are nearest neighbor, bilinear, cubic spline, etc [79].

Most importantly, wavelets is a mathematical method to decompose signals into approximate and detailed coefficients which allows the signal to be described in several levels from the coarse level (lowest resolution) to the finest level (highest resolution) [82]. Some examples of wavelets are Haar, Daubechies, Coiflets, spline, etc [83]. In general, the functionality of wavelets in 2-D is that the columns of the original image is passed through a high-pass and low-pass filter [80]. Then the rows of the filtered image are passed through the high-pass and low-pass filter [80]. If the image is transformed by another level, then the approximate coefficients will be used to transform

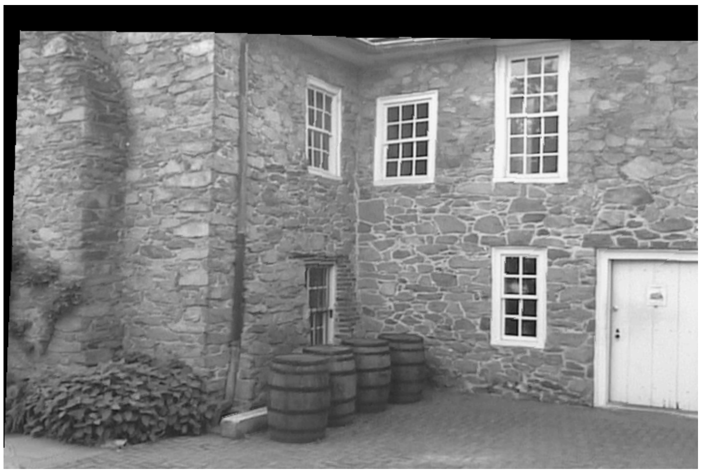


Figure 51: Our algorithm test result, in which the 37th frame is registered to the 1st frame.

the image [80]. Each pass through the filter decrease both the row and column by a multiple of two [80]. The algorithm keeps repeating these steps until the algorithm has reached n levels which is specified by the user (in our algorithm n = 4) [80]. Refer to Fig. 53 to see the process of the wavelet decomposition.

After the wavelet decomposition has been completed, the image will be divided into four subimages which are the approximate, horizontal, vertical, and diagonal. In order to obtain the approximate coefficients, the rows and columns are passed through the low-pass filter which resembles the original image, but at a smaller resolution [80]. Next the horizontal coefficients are obtained by passing the rows through the low-pass filter and the columns through the high-pass filter which will emphasize the horizontal edges [80]. Also the vertical coefficients obtained by passing the columns through the low-pass filter and the rows through the high-pass filter that will stress the vertical edges [80]. Lastly, when both the columns and rows are passed through the high-pass filter, this will produce the diagonal coefficients which accents the diagonal edges [80]. Refer to Fig. 54 to see the result of the decomposition. Now some of the benefits of using wavelets decomposition are that the important features of original image are preserved in the approximate coefficients, decrease computational speed, emphasize strong image features, and can be implemented on a parallel computer [84]. One disadvantage with wavelet decomposition is that wavelets are not shift invariant; therefore, wavelets are not able to change with the translation operator [84].

In this section, we propose a wavelet-based image registration algorithm that uses the approxi-

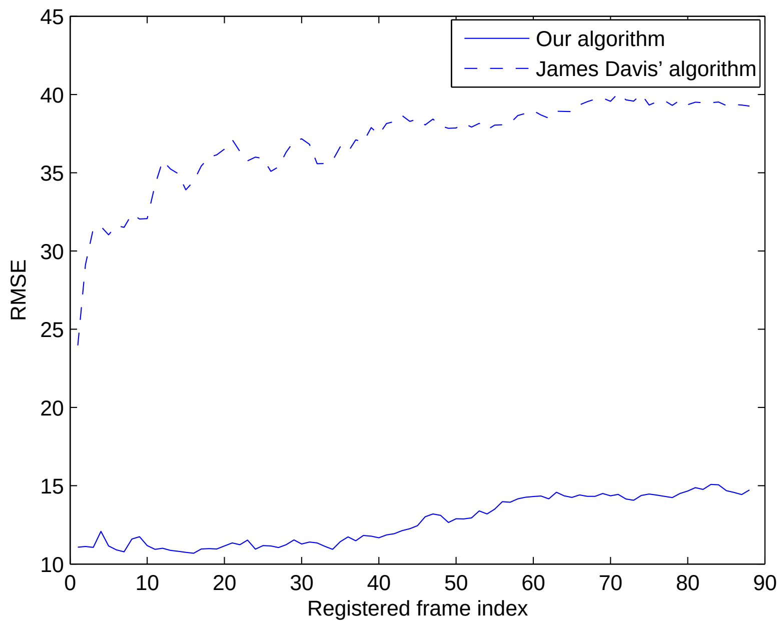


Figure 52: Our algorithm test result comparing to that under the algorithm of Davis and Keck, in which all the 88 frames are registered to the 1st frame.

mate coefficients to perform image registration. The data that we are using is the LAIR data of the CLIF2007 data set provided by Wright-Patterson Air Force Base. First the algorithm reads in the base (reference) image and input (sensed) image and removes the noise by applying the Gaussian filter. Then the filtered images are decomposed by using Daubechies wavelets and the approximate coefficients are extracted from the images. Once that is completed, the algorithm performs the gradient in both the x and y direction on the approximate coefficients to find the edges, only keeps the maximum gradient of each row in the x and y direction, and then combines all the maximum gradient of x and y onto one image which these points are the feature points in the coarse level. Next, the algorithm takes these feature points and reconstruct them onto the original size image, but now one point in the coarse level equals sixteen points in the finest level; therefore, the algorithm only keeps the points that lie on the edges of the object. Then feature correspondence is done by template matching between the input and base image by using correlation. After the correspondence is found, then RANSAC is used to eliminate the outliers to allow for better results for registration. Then the registration is performed by using projective transformation function and the resampling technique used was bicubic interpolation. Lastly the algorithm compute the Root Mean Square Error (RMSE) and the Peak Signal to Noise Ratio(PSNR). Some goals that we are trying to meet with this algorithm is to have the algorithm be fast, robust, multi-modal, automatic, RMSE below 4 intensity values, and PSNR above 35.

The remainder of this section is organized as follows. Section 9.2 shows the previous work that has been conducted with using wavelets in image registration. In Section 9.3, describes our wavelet-based image registration algorithm. Section 12.5 shows preliminary results of the algorithm.

9.2 Previous work done with wavelets in image registration

Now the section is going to discuss how wavelets has been used is other image registration algorithms, but some of the key differences from our algorithm compared to the other algorithms are that the wavelets are used only to detect features, use different wavelet transforms, similarity metrics, transformation techniques, and/or resampling methods. For example, Le Moigne, et al. [84] uses wavelet decomposition for feature selection process by computing a histogram of the horizontal (HL) and vertical (LH) coefficients for all the levels of the wavelet decomposition and saves only the points that are 13% to 15% above the maxima of the wavelet coefficients. Another image registration algorithm was created by Fonseca, et al. [85] which selects features by using the local modulus maxima of the wavelet transform and thresholding is applied on features to eliminate insignificant feature points. In order to find the correspondence this algorithm uses the maximum correlation coefficients of the approximate (LL) coefficients and utilizes the affine transformation as the transformation function. Next, Zheng, et al. [86] uses Gabor wavelet decomposition, the algorithm does feature extraction by finding the local maxima of the energy measure, uses affine as the transformation function, bilinear interpolation as the resampling technique, and feature correspondence by mutual correlation coefficients. The next algorithm is created by Li, et al. [87] which performs feature extraction by extracting a contour using a wavelet-based scheme. After feature extraction, the algorithm performs a voting algorithm on each contour point based off the intensity value which the algorithm keeps the highest score from the voting algorithm. Then the algorithm uses the normalized correlation as the similarity measure and the transform parameters are computed using the matched points which a consistency test was used to filter out mismatched points. Another example of an image registration algorithm was designed by Corvi, et al. [88] which used the residue images of the discrete wavelet transform (DWT) and clustering technique to obtained the initial transformation parameters. Also this algorithm used both the maxima and minima of the DWT coefficients to allow for more points for the feature correspondence and least mean square estimation. Next Unser, et al. [89] computed the B-Spline for the images and used the gradient-based least squares optimization in conjunction with the coarse-to-fine iteration strategy. Another image registration algorithm that used wavelets was produced by Djamdji, et al. [90], this algorithm computed the wavelets by using the algorithm à trous, feature points that are local maxima are the only points kept, compares the position of the detected feature points with that of the reference image, and then repeats the steps in the next level until the algorithm reaches the finest level. Then Quddus, et al [91] is another example of image registration algorithm which used dyadic wavelet transform as edge detection, and used mutual information in multiscale decomposition. Now Wu, et al. [92] designed an image registration algorithm that used the standard DWT due to the simplicity of the transform and in order to improve the robustness of the algorithm, the algorithm used the approximate (LL) coefficients to register images using sum of absolute differences at the lower resolution and mutual information at higher resolution. Also Wong, et al. [93] created an image registration algorithm that uses complex wavelet phase coherrence moment estimation such as Gabor and dual-tree complex wavelets for feature point detection. In order to do feature correspondence, Wong's algorithm uses normalized cross correlation between maximum complex phase coherence moments. Then the maximum distance sample consensus is used to get rid of erroneous control points and for the remaining control points, the location is adjusted iteratively to maximize the normalized cross correlation. Next algorithm created by Xishan, et al. [94] used a feature based approach to do registration by using integrated matching criteria of invariant moments and orientation of contours. In order to extract features from the images a wavelet based edge detection was used by transforming the edge strength into fuzzy field to extract well defined matchable contours. Then this algorithm performs feature correspondence by combining the invariant moment and orientation function to determine the correspondence between the contours in the images. Also Xishan chooses to use the affine transformation as the transformation function. In Bejar's, et al. [95] algorithm, the wavelets were used to extract feature points which were the edge points. Then the algorithm used normalized cross correlation to perform the feature correspondence. Next, Bejar used the moment of inertia in order to estimate the rigid transformation parameters and applied a consistency test in order to eliminate false control points. Another image registration algorithm was designed by Li, et al. [96] which decomposes the images by using the discrete wavelet frame transform which is shift invariant compared to the dyadic wavelet transform. Then Li's algorithm computed the energy map from the detail coefficients and used the genetic algorithm to obtain the minimum sum of absolute differences between the two energy maps.

Next the section is going to discuss the pitfalls of these algorithms. To start off with the algorithms that uses mutual information would significantly decrease computational speed due to mutual information being computationally expensive; therefore, this will increase the amount of time that it takes the algorithm to do registration [97]. Since one of the goals of image registration is to do image registration faster than real time, so other algorithm such as tracking can be done in real time, but the first step of tracking is to do image registration. Therefore, if image registration is not faster than real time, then tracking cannot be performed in real time, so we need to find a better way to do feature correspondence. When the algorithms uses DWT, the algorithm will not be shift invariant; therefore, these algorithms will not be robust [98]. Some wavelets that are able to be shift invariant are algorithm à trous and Gabor wavelets; however, these algorithms are computationally expensive which is undesirable since the algorithm needs to be faster than real time [98, 99]. According to Fauqueur, et al [99], we can use the Dual Tree Complex Wavelet Transform (DTCWT) to solve the shift invariant problem and has limited redundancy so this method will be less computationally expensive. Also the algorithm that uses affine transformation as the transformation function is not suitable for all images which can be seen from the results obtained from Table 7; therefore, the algorithm is not robust.

Furthermore, our algorithm is different from the other wavelet algorithms by the method in which our algorithm uses the wavelet coefficients for the feature extraction and feature correspondence, the kind of wavelet used, different similarity measures were used, and/or transformation function. Since we use DWT our algorithm will not be shift invariant; however in future work we will try other wavelets such as DTCWT to see if this improves the robustness of the algorithm. But the novelty of the algorithm will come from the future work with the ability of the algorithm to analyze when the image is registered well and when it is not. Also having the algorithm being able to determine where poor registration has occurred within the image and the potential reason for the failure which will be beneficial for defense application because it will allow other algorithm to know how much leniency to give images and know when the algorithm will not perform as well.

9.3 Wavelet-based image registration

Now we are going to discuss our wavelet-based image registration algorithm and explain each step of the image registration process.

9.3.1 Preprocessing

First step of our algorithm is to use a Gaussian filter was used in the preprocessing step to eliminate the noise in the image to improve the feature extraction. Without the Gaussian filter, the final results were approximately 5.6 for the RMSE and 33.1 for the PSNR, but with the Gaussian filter the final results were significantly improved to the RMSE being approximately 3.7 and the PSNR being approximately 36.7. The reason why there is a significant improvement in the results is because the Gaussian filter smoothed out some of the noise in the image; therefore, allowing the feature selection to select better points in both the sensed and reference image.

Now the equation for the Gaussian filter is Eq. (78). For the images we are using we set σ = 1 and the mask size of the Gaussian filter is 101$\times$101. In order to filter the image with the Gaussian filter, the image must be convoluted with the Gaussian filter Eq. (79). Where I(x,y) is the image, a = $\frac{m-1}{2}$, and b = $\frac{n-1}{2}$ [80]. Now m and n are the size of the mask and in our case m = 101 and n = 101.

$$G(x,y) = \frac{1}{2\pi\sigma^2} * \exp(-\frac{x^2+y^2}{2\sigma^2}) \tag{78}$$

$$G(x,y) \star I(x,y) = \sum_{s=-a}^{a} \sum_{s=-b}^{b} G(s,t)I(x-s,y-t) \tag{79}$$

In the future what we would like to do for the preprocessing phase is to try to filter the image using the wavelet filter. This would allow us to get rid of the Gaussian filter to save computation time since the algorithm already calculates the wavelets and the algorithm can use the information to filter the image.

9.3.2 Feature Extraction

After the sensed and reference image have been preprocessed then the images are decomposed by Daubechies 1 wavelets to the fourth level. In this algorithm, the Daub1 wavelets were used which is equivalent to the Haar wavelets, but in the future we are going to use a higher DaubJ wavelets such as Daub4. The reason for using a higher order DaubJ is because it will usually result in the signal converging faster to the original signal which means the first few levels of the detail coefficients are negligible so the approximate coefficients at the first few levels are very similar to the original image [83]. Also DaubJ multiresolution analysis produces a smoother signal than Daub1 because Daubechies wavelets have overlapping windows; therefore, Daubechies wavelets are able to detect the high frequency changes in the high frequency coefficients [83,100]. Whereas the Haar wavelet calculates the average and difference of a pair of values, then slides over two time unit, and repeats the calculation until the end of the signal is reached [100]. Therefore, causing the wavelet transform not to show the high frequency changes in the high frequency coefficients when a dramatic change occurs between an even value to an odd value [100]. However, higher DaubJ wavelets are not always the solution because sometimes a higher DaubJ will cause the wavelets to have a longer support which means that the detail coefficients are going to contain a significant amount of energy [83]. Since the detail coefficients contain a significant amount of energy, this will require more values to be used, so the data will not be able to be compressed as much [83].

One thing to note is that the term longer support refers to all the values in the wavelets that are not zero [83].

First we must get the equation for the scaling function and separable wavelet functions, so Eq. (80) is the separable scaling function, Eq. (81) - Eq. (83) are the separable wavelets. Next we need to define what the terms mean; therefore, ψ^H refers to the change along the columns which means the horizontal edges, ψ^V is the difference along the row which refers to the vertical edges, ψ^D is the variation along the diagonals, $\varphi(\cdot)$ means the 1D scaling function in the $\cdot$ direction, and $\psi(\cdot)$ stands for the 1D wavelet function in the $\cdot$ direction [80].

$$\varphi(x, y) = \varphi(x)\varphi(y) \tag{80}$$

$$\psi^H(x, y) = \psi(x)\varphi(y) \tag{81}$$

$$\psi^V(x, y) = \varphi(x)\psi(y) \tag{82}$$

$$\psi^D(x, y) = \psi(x)\psi(y) \tag{83}$$

Now that we have the separable scaling and wavelet functions, we can assign the scaled Eq. (84) and translated basis Eq. (85) - Eq. (87) [80].

$$\varphi_{j,m,m}(x, y) = 2^{j/2}\varphi(2^j x - m, 2^j y - n) \tag{84}$$

$$\psi^H_{j,m,m}(x, y) = 2^{j/2}\psi^H(2^j x - m, 2^j y - n) \tag{85}$$

$$\psi^V_{j,m,m}(x, y) = 2^{j/2}\psi^V(2^j x - m, 2^j y - n) \tag{86}$$

$$\psi^D_{j,m,m}(x, y) = 2^{j/2}\psi^D(2^j x - m, 2^j y - n) \tag{87}$$

Once we have the basis function we can now define the discrete wavelet transform for the image which can be found at Eq. (88) - Eq. (91) where M, N, H, V, D, $W_\varphi(j_0, m, n)$, $W^H_\psi(j, m, n)$, $W^V_\psi(j, m, n)$, $W^D_\psi(j, m, n)$ represents the number of columns in the image, number of rows in the the image, horizontal, vertical, diagonal, approximate coefficients at scale j_0 which is usually equal to 0, horizontal, vertical, and diagonal detail coefficients at scale j where $j \geq j_0$ respectively [80].

$$W_{\varphi}(j_0, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \varphi_{j_0,m,n}(x, y) \tag{88}$$

$$W_{\psi}^{H}(j, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \psi_{j,m,n}^{H}(x, y) \tag{89}$$

$$W_{\psi}^{V}(j, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \psi_{j,m,n}^{V}(x, y) \tag{90}$$

$$W_{\psi}^{D}(j, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \psi_{j,m,n}^{D}(x, y) \tag{91}$$

Since we use the Daub1 which is essentially the Haar wavelets, the scaling function of the Haar wavelet is Eq. (92) and wavelet function is Eq. (93) [80].

$$\varphi(x) = \begin{cases} 1 & 0 \leq x < 1 \\ 0 & otherwise \end{cases} \tag{92}$$

$$\psi(x) = \begin{cases} 1 & 0 \leq x < .5 \\ -1 & .5 \leq x < 1 \\ 0 & otherwise \end{cases} \tag{93}$$

After the wavelet decomposition of both the base (reference) and input (sensed) image, the algorithm extracts the approximate coefficients from the wavelets. The purpose for decomposing the images is to accelerate the computation speed by making the original images smaller. Once the algorithm has the approximate coefficients, then the gradient in both the x and y direction are calculated in the input and base image, and only the maximum gradient in each row is saved because that is where an edge point should lie since that should be an area of greatest change. In order to see the equation for the gradient, refer to Eq. (94) which is the rate of change in the x and y direction [80]. Next, the maximum gradients in the x and y direction are combined together for both the base and input image. Then the algorithm create two black images the same size as the coarse image, finds the location of the control points, extracts the corresponding approximate coefficients, places the approximate coefficients into the black images, and the images are reconstructed to the original size. Now the reconstructed image provides the algorithm with a mask to tell where the control points are located on the original image. One problem is that one point in the coarse image is equal to 16×16 points in the finest level, since the algorithm went to the fourth level of the wavelets. To fix this problem, for each 16×16 control point, the algorithm takes the absolute difference of each column of the control point, selects the n greatest difference of each column to become the control points, and n = 10 in our algorithm. These points should be the points that lie on the edge of the object because that is where the greatest change would most likely occur.

$$\nabla(f) \equiv grad(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} \equiv \begin{bmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{94}$$

However, there are some issues that need to be addressed in the future work. One example of what we are going to try in the future work is to see if the algorithm will improve by using the Horizontal, Vertical, and Diagonal coefficients rather than using the gradient method to save some computational time. Another issue I see with this feature extractor is the control points could potentially not be spread throughout the image which would have a negative effect on the outcome of the image registration. Some examples of such an issue is that you have a body of water running through the image which will not pick up significant features or the image has a black and white line next to each other in the image, so the greatest change would be along that line which will produce horrible results because the points will be concentrated in a certain region of the image.

9.3.3 Feature Correspondence

Once the algorithm has extracted the feature points, the algorithm must determine which feature points in the input image matches with the feature points in the base image. First, the points along the edges of the image are discarded because there is a w×w window (in our algorithm w = 9) that scans through the images to get a template to do the correspondence, so if we kept the points that were on the edge of the image then the algorithm would crash. Therefore, the region that will be scanned is $\frac{w}{2}$ through m-$\frac{w}{2}$ and $\frac{w}{2}$ through n-$\frac{w}{2}$, where m and n are the number columns and rows of the image respectively. Next the algorithm must round down $\frac{w}{2}$ because w needs to be an odd number so the control point can be located exactly in the middle of the window which when we divide an odd number by 2 we will have a decimal. Then one control point in the base image is selected, a w×w window is created around the control point which is considered the middle of the w×w window, and now there is template of a w×w window of intensity values of the base image. Once the template is extracted from the base image, then the template is compared to all the control points in a b×b window (in our algorithm b = 500) of the input image and b needs to be much greater than w. Before the comparison is done between the base and input control points, a w×w window is created around the input control point with the control point being the middle of the window. Next, the comparison is done by performing the wavelet decomposition on the template, extracting the approximate coefficients, and performing correlation based off of the normalized dot product, refer to Eq. (95) to see the equation. Now the variables are tb, ti, and i which refer to the base template, input template, and the location in the w×w window, respectively. After the comparison between the two points is done, then the base control point is compared to another input control point within the b×b window, if the correlation between those two points is higher than the current maximum correlation, then that point is saved as the point with the highest correlation. Then comparing the base feature point to all the input feature points in a b×b window is done until all the input feature points within the window have been examined. Once all the input feature points have been examined, then input feature point which has the highest correlation and is above a certain threshold (our algorithm the threshold is .995) is considered the corresponding point to the base feature point. Then the algorithm goes to the next base control point and repeats the previous steps to find the corresponding input control point. The algorithm performs the feature correspondence until all the base control points have been examined. To further improve the correspondence of the algorithm, Random Sample Consensus (RANSAC) was used to get rid of the outliers. In order to have a further understanding how RANSAC works, please refer to the section by Fischler [101].

$$corr = \frac{\sum_{i=1}^{l} tb_i \cdot ti_i}{\sqrt{\sum_{i=1}^{l}(tb_i \cdot tb_i) \sum_{i=1}^{l}(ti_i \cdot ti_i)}} \tag{95}$$

One of the biggest issues with this algorithm is that the computation time takes too long, so improvement needs to be done in this area because the computation time takes about an hour for one pair of images. An idea is to do correspondence in the coarse level, then go to the next finest level, perform correspondence just on the points that the algorithm deemed as correspondence points, and do this until the finest level has been reached which has been done in other algorithms. This should speed up computation speed considerably because the majority of the computation will be done in the coarse level which is significantly smaller than the original image. Another idea that we may try is to continue with correspondence in the coarse image, but split the image into section, find a certain number of points in each section, and once number has been reached then continue onto the next region. These are some ideas that we are going to try in our future work.

9.3.4 Transformation

After Feature Correspondence the algorithm uses the projective transformation because the lens and sensor nonlinearities do not exist and the scene is relatively close to the camera [79]. Also we tried the other transformation function such as linear conformal, affine, and polynomial, but projective gave us the best results which was based off the RMSE and PSNR (Refer to Table 7.

The equations for the projective transformation are Eq. (96) and Eq. (97). Now a through h are the eight unknowns that can be solved by at least four non-colinear corresponding points in the images [79]. Usually the number of control points that are used are more than the minimum required amount of control points which usually allows for better results at the cost of requiring more computation time to determine feature correspondence of the points [77, 78].

$$X = \frac{ax + by + c}{dx + ey + 1} \tag{96}$$

$$Y = \frac{fx + gy + h}{dx + ey + 1} \tag{97}$$

9.3.5 Resampling

Following the transformation step, The algorithm uses the bicubic interpolation to do the resampling of the image. In our algorithm, bicubic interpolation was used because it produces a more accurate result than nearest neighbor or bilinear interpolation but does require more computational time [79]. Also refer to Table 8 for actual results obtained by our algorithm.

9.3.6 Algorithm Summary

In order to see a general overview of how our algorithm functions, refer to Fig. 55. Then look down below in Table 6 which shows step by step how the algorithm performs image registration, but for details for each step look at Section 9.3.1 through Section 9.3.5.

Table 6: Wavelet-Based Image Registration

• **Preprocessing:** First, read in both the base and input image and then filter both the base and input image using Gaussian Filter.
• **Feature Extraction:** After the image has been preprocessed, then apply wavelet transformation on both the base and input images using Daubechies Method (db1 or haar) to the fourth level. Next, extract the approximate coefficients from the fourth level and calculate the gradient of the base and input approximate images in both the x and y direction. After finding the gradient of both images, then find the maximum gradient in both the x and y direction for the base and input images and combine max gradient of the x and y direction onto one image for both images. Then reconstruct the base and input energy images into the original image size. Once energy images have been reconstructed to the original size, the next step is to map the location of the control points on the energy images onto the actual image. Since one point in the coarse level equals 16×16 in the finest level; therefore, the algorithm filters out all the unnecessary point by picking the points that lie on the edges in the finest level.
• **Feature Correspondence:** After getting the control points, then find correspondence between base and input control points using correlation. Then use RANSAC to eliminate the outliers to further improve the registration results.
• **Transformation Function:** Once RANSAC is completed the algorithm uses Projective transformation as the transformation function.
• **Resampling:** The last step of the registration algorithm is to perform the resampling technique by using bicubic interpolation.

Table 7: Results of using Different Transformation Techniques

	Linear Conformal	Affine	Projective	Polynomial
PSNR	23.4397	30.6937	36.5214	36.1649
RMSE	17.1614	7.4449	3.8060	3.9655

Table 8: Resulting using Different Resampling Technique

	Nearest Neighbor	Bilinear	Bicubic
RMSE	4.2286	3.8654	3.8620
PSNR	35.6068	36.3868	36.3945

9.4 Results

In order to see how well our algorithm performed we simulated a preliminary test which produced desirable results with the RMSE being 3.7361 and the PSNR being 36.6824 (our goal of RMSE being below 4 and PSNR being above 35 was met) (Fig. 56). One reason for using the RMSE is because the Mean Square Error (MSE) is a multiple of the energy of the difference between the images. Now RMSE is the square root of the MSE which the RMSE will allow us to compare the data as the intensity values whereas MSE would be (intensity value)^2. Also we used PSNR as a value measure because the PSNR is a logarithmic measure which studies have shown that our brain reacts logarithmically to alteration in light intensity [83]. Next we explain what the variable mean in the equation below, so P in equation Eq. (98) is the total number of pixels used in the calculation, i is the column location of the pixel, and j is the row location of the pixel. Since this is only tested between one pair of images, there is still much work that needs to be done with testing of the algorithm on the rest of the data set, but the speed of the algorithm needs to improve significantly before we can test it on the other data sets.

$$MSE = \frac{1}{P}\sum_{i,j}(f_{i,j} - g_{i,j})^2 \tag{98}$$

$$RMSE = \sqrt{MSE} \tag{99}$$

$$PSNR = 10log_{10}\frac{255^2}{MSE} \tag{100}$$

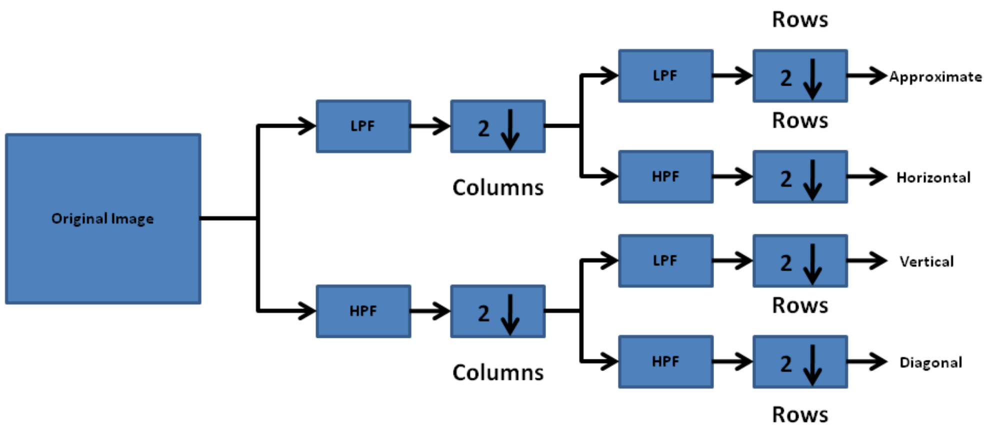


Figure 53: 2-D Wavelet Transform

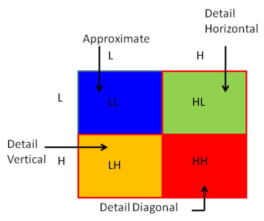


Figure 54: Wavelet Decomposition

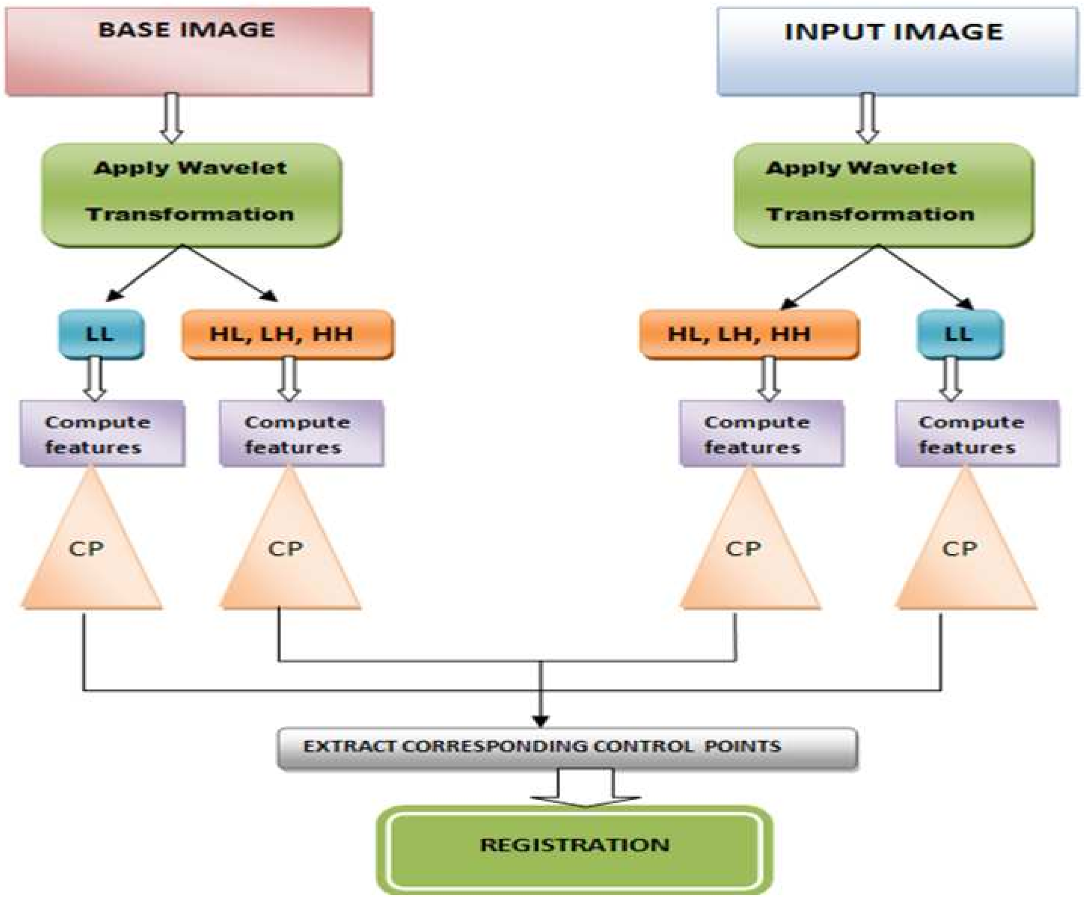


Figure 55: Flow Chart of the Wavelet Based Image Registration Algorithm

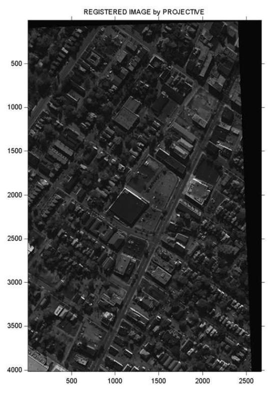


Figure 56: Registered Image

10 3D Surface Recovery via Deterministic Annealing based Piecewise Linear Surface Fitting Algorithm

The objective of Task 8 is to develop an algorithm to automatically recover 3D surface from sparse 3D points.

The 3D surface fitting problem is to find a 3D surface that fits to a set of 3D points. Geometric fitting is commonly used in computer vision for 3D modeling and reconstruction. Finding a good fit to a given data set is a classical and challenging problem. Although there are many existing algorithms for specific cases, the geometric fitting problem is far from 'solved'. Geometric fitting is highly related to statistical regression, which is to approximate an unknown mathematical function that fits to input-output data pairs observed with random errors. In this paper, we present a new piecewise plane fitting method for 3D surface fitting. Different from traditional algorithms, we first segment the input space to several separate regions. With an assumption that each region is locally linear, we can use plane fitting to recover the 3D geometric surface. We propose a non-linear deterministic annealing algorithm for space partitioning. The non-linear deterministic annealing algorithm is able to avoid many shallow local optima. The algorithm also considers local structures to help data partitioning In the optimization process. The experimental results show that the new method can achieve better performance in both the average approximation error and correct identification rate on both synthetic data and real world data.

The results of Task 8 were reported in Ref. [102]. Next, we present the technical details.

10.1 Introduction

The geometric fitting problem is to find a geometrical surface that best fits to a set of 3D points. Geometric fitting is commonly used in 3D model fitting and 3D visual reconstruction in computer vision. The 3D surface fitting is highly related to statistical regression, which is an important tool in diverse areas.

Given a 3D point data set $\mathfrak{X} = \{\mathbf{x}_i\}, \mathbf{x}_i \in R^3, i = 1, 2, ..., n$, the geometrical fitting problem is usually stated as the optimization of a cost that measures how the geometrical surface function $\mathfrak{G} = \{\mathbf{x} : g_\theta(\mathbf{x}) = 0\}$ fits the data set $\mathfrak{X}$. The most commonly used objective function is the least squares cost,

$$D = \sum_{i=1,...N} d(\mathbf{x}_i, g_\theta)^2 \tag{101}$$

where

$$d(\mathbf{x}_i, g_\theta) = \min \|\mathbf{x}_i - \mathbf{x}_j\|^2, \;\; \mathbf{x}_j \in \mathfrak{S} \tag{102}$$

The fitting function g_θ is learned by minimizing the design cost, D, measured over the input data set, $\mathfrak{X}$. It is well-known that for most choices of D, the cost measured during design monotonically decreases as the size of the learned fitting function g_θ is increased. With a large set of functions, it is easy to create a surface which passes through each input data point but is suspiciously complicated. The principle of Occam's razor states that the simplest model that accurately represents the data is most desirable. So we prefer to use a few basis functions which yield a smoother, simpler surface which could well approximates the original data.

Generally, there are two approaches to solve the over fitting problem. One approach is to add penalty terms to the data set, like smoothness or regularization constraints. Another approach is to first build a large model and then remove some parameters by retaining only the vital model structure. Although both approaches can generate parsimonious models, the descent based learning methods all suffer from a serious limitation. The non-global optima of the cost surface may easily result in poor local minima to the descent based learning methods. Techniques adding penalty terms to the cost function further increases the complexity of the cost surface and worsen the local minimum problem.

In this section, we propose a different approach to solve the geometrical fitting problem. Instead of estimate a complicated function to fit all the data points, we partition the data set into several subset such that the data points in each subset could be approximated by a simpler model. The space partitioning helps to reduce the size of the surface model while keeping the design cost small enough.

One of the most popular clustering algorithm is Lloyd's algorithm, which starts by partitioning the input data into k initial sets. It calculates the centroid of each set via some metric. Usually, Lloyd's algorithm is used in a Euclidean space and centroid is calculated by averaging dimensions in Euclidean space. It iteratively associates each point with the closest centroid and recalculates the centroids of the new clusters. Alghouth widely used in real world applications, there are two serious limitations of Lloyd's algorithm. The first limitation is that the partitioning result depends on the initialization of the cluster centers, which may lead to poor local minima. The second limitation is that Lloyd's algorithm can only partition linear separable clusters.

In order to avoid initialization dependence, a simple but useful solution is to use multiple restarts with different initializations to achieve a better local minima. Global k-means [103] is proposed to build the clusters deterministically, which use the original k-means algorithm as a local search step. At each step, global k-means add one more cluster based on previous partitioning result. Deterministic annealing [104] is another optimization technique to find a global minimum of a cost function. Deterministic annealing explore a larger cost surface by introducing a constraint of randomness. At each iteration, the randomness is constrained and a local optimization is performed. Finally, the imposed randomness is reduce to zero, and the algorithm optimizes over the original cost function.

Kernel method [105] is used to solve the second problem by mapping the data points from input space to a higher dimensional feature space through a non-linear transformation. Then the optimization is applied in the feature space. The linear separation in the feature space turns out to be a non-linear separation in the original input space.

In this section, we propose a non-linear deterministic annealing approach for space partitioning in 3D Euclidean space. We use deterministic annealing to divide the input space into several regions with different sizes and shapes. With the partition, we can easily find a linear local surface to fit the data inside each region. Deterministic annealing method offers two great features: 1) the ability to avoid many poor local optima; 2) the ability to minimize the cost function even its gradients vanish almost everywhere. Due to the fact that the data is localized to a few relatively dense clusters, we design a kernel function to map the data point from the geometric space to surface feature space and apply deterministic annealing in the feature space instead of the geometric space. We compare the proposed non-linear deterministic annealing (NDA) algorithm with the widely used Lloyd's algorithm on both artificial data and real world data. The experimental results show that NDA algorithm outperforms Lloyd's algorithm in both mean squared approximation error and error probability.

In the following section we formally define the 3D geometric fitting problem and briefly describe deterministic annealing and kernel method for space partitioning. In Section 11.4.2 we present the proposed kernel deterministic annealing algorithm along with an analysis of its computational complexity. The experimental result is shown in Section 12.5.

10.2 The 3D Geometric Fitting Problem

The following terms and notations are used throughout this report.

- Input samples $\mathbf{x}_1, ..., \mathbf{x}_N \in \mathcal{R}^3$ are 3D data points.
- A feature vector $\mathbf{f}_i = \Phi(\mathbf{x}_i) \in \mathcal{F}$ is computed by some mapping $\Phi : \mathcal{R}^3 \to \mathcal{F}$. It typically consists of a vector of d measurements: $\mathbf{f} = (f_1, ..., f_d)$.
- d is the dimensionality of the pattern or of the pattern space.

- A data set is denoted $\mathfrak{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$.

Given a set of data $\mathfrak{X}$ of scattered 3D points, we would like to find the geometric surface that best fits to the scattered data. The fitting problem is usually stated as the optimization of a cost that measures how well the fitting function $g(\mathbf{x}_i)$ fits the data. The most commonly used objective function is the least squares cost. Finding a good fit is a challenging problem and may be more of an art than a science. If we use a large set of functions as the basis, we may create a surface which passes through each data point but is suspiciously complicated. Using few basis functions may yield a smoother, simpler surface which only approximates the original data. Due to the over fitting problem, we propose an new approach to optimize the objective function via space partitioning. We first partition the data set into several subsets such that the data points $\mathbf{x}$ in each subset could be approximated by a linear surface model. In other words, we would like to use a set of plain models to approximate the date set. The objective of space partitioning is to minimize the geometric fitting error.

$$\min_{g_{\theta_k}} D = \sum_{k=1}^{K} \sum_{i \in C_k} d(\mathbf{x}_i, g_{\theta_k}) \tag{103}$$

where, $\mathbf{x}_i = [x_i, y_i, z_i]^T$ is the i-th point data, $\theta_k = [a_k, b_k, c_k]^T$ is the k-th linear surface model, and $d_{i,k}$ is is the fitting error between $\mathbf{x}_i$ and plane model $g_{\theta_k} = 0$ which is defined as

$$d_{i,k} = d(\mathbf{x}_i, g_{\theta_k}) = \frac{(\mathbf{x}_i^T g_{\theta_k} - 1)^2}{a_k^2 + b_k^2 + c_k^2} \tag{104}$$

10.2.1 Deterministic Annealing

The deterministic annealing (DA) approach [104] to clustering has demonstrated substantial performance improvement over traditional supervised and unsupervised learning algorithms. DA mimics the annealing process in static The advantage of deterministic annealing is its ability to avoid many poor local optima. The reason is that deterministic annealing minimizes the designed cost function subject to a constraint on the randomness of the solution. The constraint, Shannon entropy, is gradually lowered and eventually deterministic annealing optimize on the original cost function. Deterministic annealing mimics the simulated annealing [106] in statistical physics by the use of expectation. Deterministic annealing derives an effective energy function through expectation and is deterministically optimized at successively reduced temperatures. The deterministic annealing approach has been adopted in a variety of research fields, such as graph-theoretic optimization and computer vision. A. Rao et al. [107] extended the work for piecewise regression modeling. In this subsection, we will briefly review their work.

Given a data set $(\mathbf{x}, \mathbf{y})$, the regression problem is to optimize the cost that measures how well the regression function $f(\mathbf{x})$ approximates the output $\mathbf{y}$, where $\mathbf{x} \in \mathcal{R}^m$, $\mathbf{y} \in \mathcal{R}^n$, and $g : \mathcal{R}^m \to \mathcal{R}^n$. In the basic space partitioning approach, the input space is partitioned into K regions and the cost function becomes

$$\min_{\mathbf{\Lambda}_k} D = \sum_{k=1}^{K} \sum_{i \in C_k} d(\mathbf{y}_i, f(\mathbf{x}_i, \mathbf{\Lambda}_k)) \tag{105}$$

where $d(\cdot, \cdot)$ is the distortion measure function. Instead of seeking the optimal hard partition directly, randomness is introduced for randomized assignment for input samples.

$$D = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) d(\mathbf{y}_i, f(\mathbf{x}_i, \mathbf{\Lambda}_k)) \tag{106}$$

In A. Rao et al.'s work, they use the nearest prototype (NP) structure as constraint and given the set of prototypes$\{\mathbf{s}_j : j = 1, 2, 3, ..., K\}$ in the input space, a Voronoi criterion is defined for NP partition

$$C = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) ||\mathbf{x}_i - \mathbf{s}_j||. \tag{107}$$

Although the ultimate goal is to find the hard partition, some "randomness" is desired during the assignment. Shannon entropy is introduced as a constraint of the randomness.

$$H = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) \log P(\mathbf{x}_i \in C_j). \tag{108}$$

Eventually, this constrained optimization problem could be rewritten as the minimization of the corresponding Lagrangian

$$\min_{\{\mathbf{\Lambda_j}\}\{\mathbf{s_j}\},\gamma} F = D - TH \tag{109}$$

where, γ is a nonnegative Lagrange multiplier which controls the randomness of the space partition.

10.2.2 Non-linear Partitioning

Kernel methods (KMs) are a class of algorithms for pattern analysis whose general task is to find and study types of relations of input data. KMs perform a nonlinear mapping of the input data to a higher dimensional feature space. Then a variety of methods can be applied for pattern analysis in the feature space. The advantage of KMs is that KMs do not need to compute the coordinates of the data in the feature space explicitly but only compute the innor products between all pairs of data in the feature space by using kernel functions.

Take the most popular k-means algorithm [108] as an example, kernel k-means maps data points from the input space to a higher dimensional feature space through a nonlinear transformation ϕ and then apply standard k-means in the feature space. The clustering result in linear separators in feature space corresponds to nonlinear separators in input space. Thus kernel k-means avoid the limitation of standard k-means that the clusters must be linearly separable.

10.3 Non-linear Deterministic Annealing

In this section, we propose a new approach based on non-linear deterministic annealing to solve the 3D geometric fitting problem. We first use a non-linear function to map the input point data to a high dimensional feature space using the local geometric structure of the data. Then we apply deterministic annealing in the feature space to leverage the local geometric structure for clustering.

To solve the space partitioning problem, we do not use prototype to calculate the difference. The reason is that the prototype in space partitioning is generally not sufficient to represent a plane in 3D space. Instead, we estimate the linear plane model and calculate the fitting error as the Euclidean distance between the data and the plane. The traditional local optimization algorithm will likely stuck at a local optima. In order to avoid local optima, we use local geometric structure from neighboring data points and embedded the data vectors to a higher dimension as follows.

The input data is given as a 3D point, $\mathbf{x}_i = [x_i, y_i, z_i]^T$. With the assumption that nearest data points are on the same plane, we could estimate the local plane model, $\mathbf{L}_i = [a_i, b_i, c_i]^T$ of data point $\mathbf{x}_i$ and its K nearest neighbor points.

$$\mathbf{L} = \begin{bmatrix} a(\mathfrak{X}) \\ b(\mathfrak{X}) \\ c(\mathfrak{X}) \end{bmatrix} \tag{110}$$

$$\mathbf{f} = \begin{bmatrix} \mathbf{x} \\ \mathbf{L} \end{bmatrix} \tag{111}$$

Then we revise the distortion function as follows,

$$D(\mathbf{f}_i, g_{\theta_j}) = D_1(I_1\mathbf{f}_i, g_{\theta_j}) + D_2(I_2\mathbf{f}_i, g_{\theta_j}) \tag{112}$$

$$I_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{113}$$

$$I_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{114}$$

where $D_1 = d_{i,j}$ calculate the fitting error between the data point and the estimated plane, and D_2 calculate the difference between the local estimated plane model and the cluster scale estimated plane model. D_2 is defined as follows:

$$D_2(I_2\mathbf{f}_i, g_{\theta_j}) = \frac{I_2\mathbf{f}_i^T \times g_{\theta_j}}{|I_2\mathbf{f}_i| \times |g_{\theta_j}|} \tag{115}$$

After the mapping, we apply deterministic annealing algorithm to partition the data into several clusters as follows.

$$\min_{g_{\theta_j}} F = D - TH \tag{116}$$

where $g_{\theta_j} = [a_j, b_j, c_j]$ is the geometrical surface model parameter to be estimated, D is the sum of square of geometrical fitting error and H is the entropy constraint. We define D and H as follows:

$$D = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j})d(\mathbf{x}_i, g_{\theta_j}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i)d(\mathbf{x}_i, g_{\theta_j}) \tag{117}$$

$$H(\mathbf{X}, g_\theta) = \sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j}) \log p(\mathbf{x}_i, g_{\theta_j}) \tag{118}$$

To perform optimization we need to further analyze its terms. We can rewrite equation (186) by applying the chain rule of entropy as

$$H(\mathbf{X}, g_\theta) = H(\mathbf{X}) + H(g_\theta|\mathbf{X}) \tag{119}$$

Notice that the first term $H(\mathbf{X})$ is the entropy of the source and is therefore constant with respect to the cluster g_{θ_j} and association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$. Thus we can just focus on the conditional entropy

$$H(g_\theta|\mathbf{X}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i) \log p(g_{\theta_j}|\mathbf{x}_i) \tag{120}$$

The minimization of F with respect to association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$ gives rise to the Gibbs distribution

$$p(g_{\theta_j}|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T})}{Z_x} \tag{121}$$

where the normalization is

$$Z_x = \sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T}) \tag{122}$$

The corresponding minimum of F is obtained by plugging equation (189) back into equation (184)

$$F^* = \min_{p(g_{\theta_j}|\mathbf{x}_i)} F = -T \sum_{i=1}^{N} p(\mathbf{x}_i) \log Z_x \tag{123}$$

To minimize the Lagrangian with respect to the cluster model g_{θ_j}, its gradients are set to zero yielding the condition

$$\nabla_{g_{\theta_j}} F = \frac{1}{N} \sum_{i=1}^{N} p(g_{\theta_j}|\mathbf{x}_i) \nabla_{g_{\theta_j}} d(\mathbf{x}_i, g_{\theta_j}) = 0 \tag{124}$$

Non-linear deterministic annealing method (NDA) introduces the entropy constraint to explore a large portion of the cost surface using randomness, while still performing optimization using local information, which is similar to fuzzy c-means algorithm. Eventually, the amount of imposed randomness is lowered so that upon termination NDA optimizes over the original cost function and yields a solution to the original problem.

However, there is no close form solution for NDA, therefore we use a gradient descent algorithm to solve this problem. In this section, We compare NDA based geometrical segmentation algorithm to the projection based iterative algorithm (PI) and adaptive projection based iterative algorithm (API). I present our algorithm in Figure. 78. For comparison purpose, I also give PI algorithm in Figure. 58.

```
1.  Algorithm 78 **NDA based geometrical segmentation algorithm**
2.  **Set Limit**
3.      $K_{max}$: maximum number of clusters
4.      $T_{init}$: starting temperature
5.      $T_{min}$: minimum temperature
6.      $\delta$: perturbation vector
7.      $\alpha$: cooling rate (must be $< 1$)
8.      $I_{max}$: maximum iteration number
9.      $th$: Iteration threshold
10.     $sth$: Surface distance threshold
11. **Initialization**
12.     $T = T_{init}, K = 2, \Lambda_1 = (X^T X)^{-1} X^T \vec{1}, \Lambda_2 = \Lambda_1, [p(\mathbf{\Lambda}_1|\mathbf{x}_i), p(\mathbf{\Lambda}_2|\mathbf{x}_i)] = [\frac{1}{2}, \frac{1}{2}], \forall i.$
13. **Perturb**
14.     $\Lambda_j = \Lambda_j + \delta, \forall j.$
15.     $L_{old} = D - TH.$
16. **Loop until convergence,** $i = 0 \; \forall j$
17.     For all $\mathbf{x}_i$ in the training data, compute the association probabilities
            $p(\mathbf{\Lambda}_j|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i,\mathbf{\Lambda}_j)}{T})}{\sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i,\mathbf{\Lambda}_j)}{T})}$    (125)
18.     update the surface model
            $\mathbf{\Lambda}_j \longleftarrow \mathbf{\Lambda}_j + \alpha \nabla_{\mathbf{\Lambda}_j} F.$    (126)
19.     i = i+1;
20.     if ($i > I_{max}$ or $\nabla_{\mathbf{\Lambda}_j} F < th$ ) End **Loop**
21. **Model Size Determination**
22.     if($d(\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}) < sth$)
23.         replace $\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}$ by a single plane
24.     $K$ =number of planes after merging
25. **Cooling Step**
26.     $T = \alpha T.$
27.     if ($T < T_{min}$)
28.         perform last iteration for $T = 0$ and STOP
29. **Duplication**
30.     Replace each plane by two planes at the same location, $K = 2K$.
31. **Goto Step 10**
```

```
1. Algorithm 58 **Projection based iterative algorithm for geometrical segmentation**
2. **Set Limit**
3.    $K_{max}$: maximum number of clusters
4.    $I_{max}$: maximum iteration number
5.    $th$: Iteration threshold
6. **Initialization**
7.    Start with a random cluster assignment to all input vectors and estimate the linear plane model for each
   cluster by minimizing the total least squares.
8. **Loop until convergence,** $i = 0\ \forall j$
9.    a. Assign each input vector $\mathbf{x}$ to the each cluster with the smallest geometrical fitting error.
10.   b. Estimate the linear plane model for each cluster by minimizing the total least squares.
```

Figure 58: Projection based iterative algorithm for geometrical segmentation

10.4 Experimental Results

In this section, I compared three geometric segmentation algorithms, PI algorithm, API algorithm, and NDA based geometric segmentation algorithm, based on both synthetic data and real world data.

10.4.1 NDA on Synthetic Data without Noise

The purpose of the first experiment is to compare NDA, PI, and API on synthetic data without noise. I generated the synthetic data using MATLAB ‘randperm’ function. The data is a set of 3D points on several linear planes without noise. In this experiment, I run each algorithm for 1000 times. Each time, a random data set is generated and used. We segment the same data set with different algorithms and calculate the average squared approximation error. Below is the experimental result in Table. 12. K represents the number of planes in a test data set. For each plane, 100 random points are generated. The date set 1 contains 300 data in total from 3 non parallel planes. The data set 2 contains 400 data from 4 planes. The data set 3 contains 500 data from 5 planes and the data set 4 contains 600 data from 6 planes. The average squared approximation error of NDA is ignorable comparing to the errors of PI and NPI. From the experimental result, we can say that NDA algorithm outperforms both PI and API algorithms in the average squared approximation error. The reason NDA algorithm outperforms PI and API algorithms is that NDA is able to separate the space non-linearly and avoid many poor local optima.

Table 9: The average squared approximation error.

K	PI	API	NDA
3	3.77×10^{-1}	3.00×10^{-9}	1.17×10^{-12}
4	4.01×10^{-1}	9.81×10^{-8}	2.21×10^{-12}
5	2.43×10^{-1}	2.86×10^{-9}	3.06×10^{-12}
6	2.94×10^{-1}	8.801×10^{-9}	3.00×10^{-12}

Table 10: The correct identification rate.

K	PI	API	NDA
3	83%	96%	99%
4	79%	93%	99%
5	82%	94%	97%
6	78%	97%	98%

We also measure the performance of the segmentation algorithms in percentage of correct identification of planes. We test the same data set as used in the previous experiment and compute the correct identification percentage averaging over all tests. Below is the experimental result in Table. 13. We observed that correct identification rates of NDA and API are much higher than the correct identification rate of PI algorithm. The reason API algorithm outperforms PI algorithm is that API algorithm does not depends on random initialization while the segmentation results of PI algorithm heavily depends on initialization. Still NDA performs best among the three algorithms in correct identification rate.

10.4.2 NDA on Synthetic Data with Noise

The purpose of the second experiment is to compare NDA, PI, and API algorithms on synthetic data with noise. I generated the synthetic data with Gaussian noises in the same way as in the first experiment. In this experiment, I also run each algorithm for 1000 times. Each time, a random data set is generated and used. We segment the same data set with different algorithms and calculate the average squared approximation error. The experimental result is shown in Table. 11. K represents the number of planes in a test data set. It shows that NDA algorithm outperforms both PI and API algorithm. The average squared approximation fitting error of NDA algorithm is less than 50% compare to the fitting error of PI algorithm. However, the performance gain is less compared to

Table 11: The average squared approximation error.

K	PI	API	NDA
3	6.61×10^{-1}	8.96×10^{-1}	2.41×10^{-1}
4	8.18×10^{-1}	5.98×10^{-1}	3.19×10^{-1}
5	6.98×10^{-1}	4.42×10^{-1}	3.96×10^{-1}
6	1.16	9.44×10^{-1}	6.71×10^{-1}

the first experiment. The reason is that the non-linear mapping in NDA depends on the estimation of the local geometric structures. While the estimation of the local geometric structures is very sensitive to the added noises. Even though the performance gain is less, we can still say that the NDA algorithm outperforms both PI and API algorithms in the average squared approximation error from the experimental result. We also show the experimental result in 3D view in Fig. 59 and Fig. 60. Fig. 59 shows the segmentation results of test data set 1 with three planes by the NDA algorithm. Fig. 60 shows the segmentation results of the same test data set by the PI algorithm.

10.4.3 NDA on Real World Data

In the second experiment, we test the geometric segmentation algorithm on some real world data. We use the 3D structure data set from the ‘housing’ image sequence. The data set includes 72 data points recovered by 3D reconstruction of 2D registered feature points. Most of the data points fall on the walls of the house in the image and we would like to estimate the surface model of the walls by geometric fitting. Fig. 67 shows the input 3D data points on the 1st frame of the ‘housing’ image sequence. The goal is to segment the data points into three groups and each group represent a wall in the image. Fig. 68 shows the geometric segmentation result by NDA algorithm and Fig. 69 shows the geometrical segmentation result by PI algorithm. It is pretty clear that NDA algorithm partitions the input data set into three clusters and each cluster represents a wall in the image. PI algorithm fails to find the geometric model of the walls and the data points are mixed. The experimental result on real world data shows that NDA algorithm can well segment the data sets based on their geometric relationship.

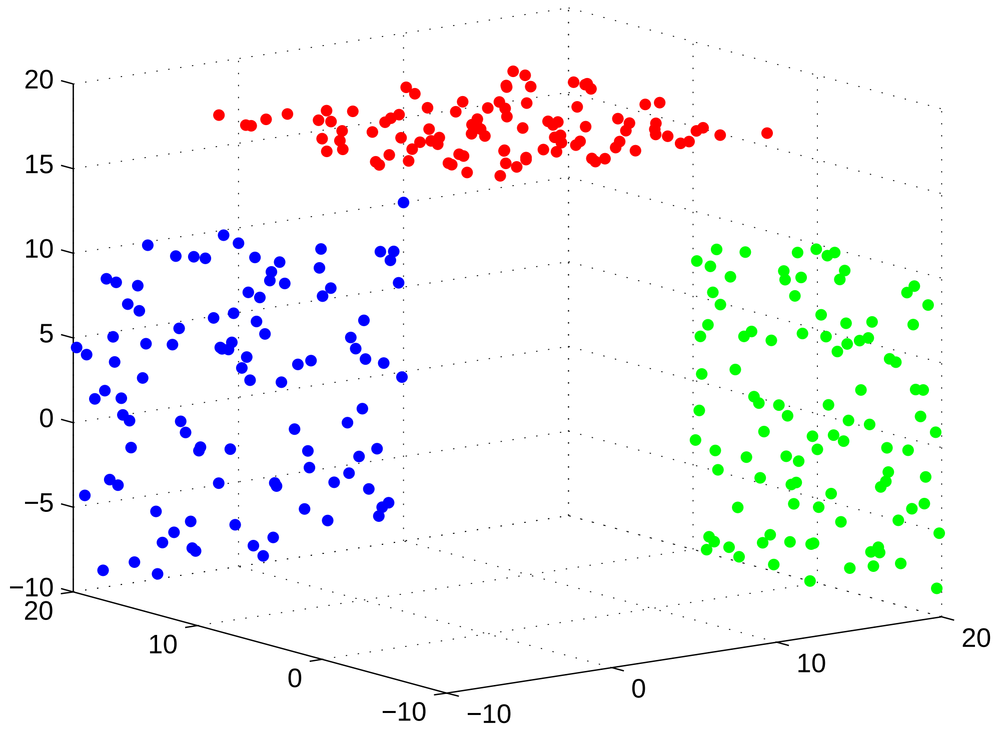


Figure 59: The synthetic data set.

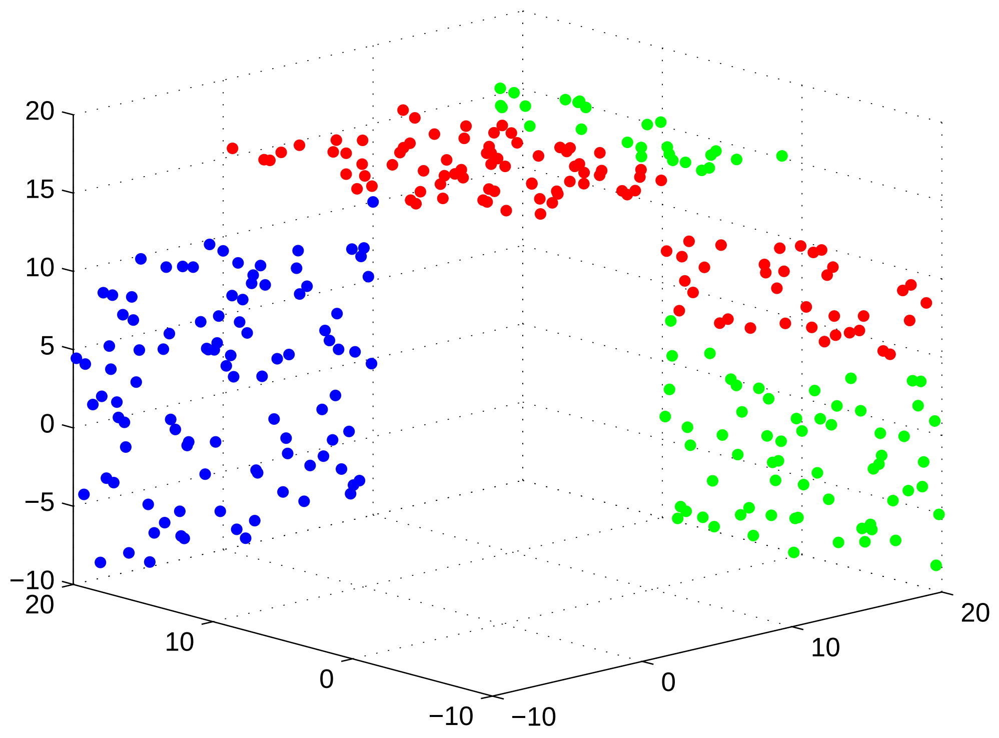


Figure 60: The first group partitioned by K-means.

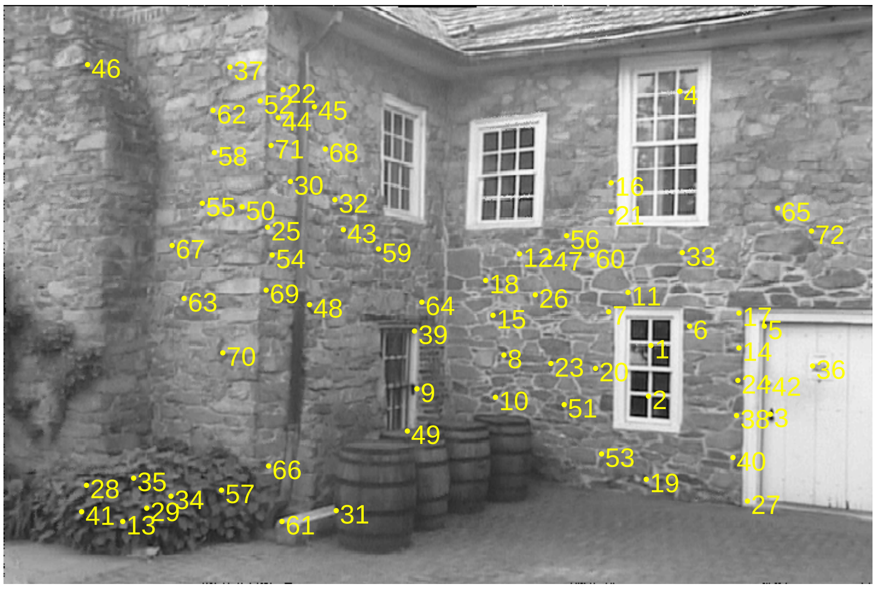


Figure 61: The input data points on the 1st frame of 'housing' image sequence.

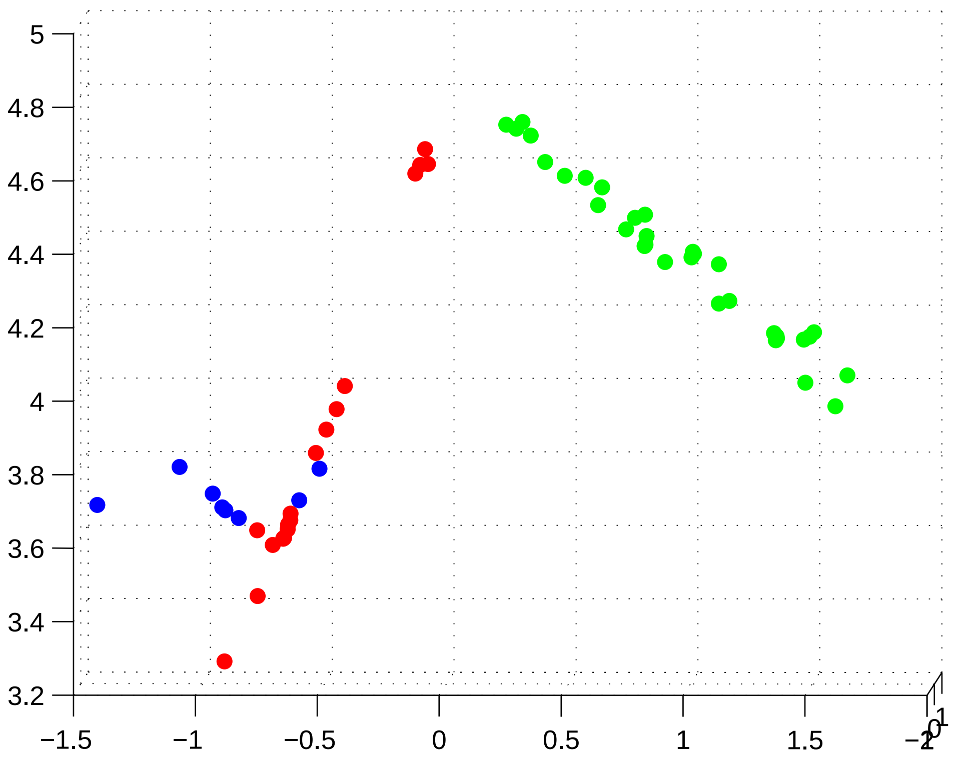


Figure 62: The geometrical segmentation result by NDA of ‘housing’ data set.

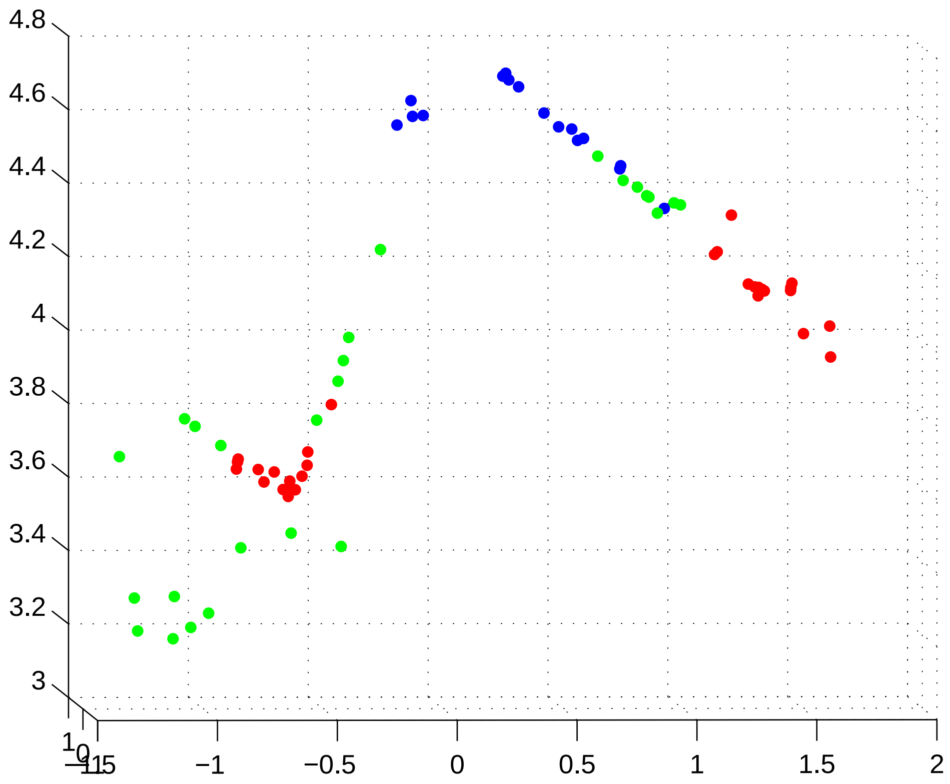


Figure 63: The geometrical segmentation result by the PI algorithm of ‘housing’ data set.

11 An Automatic Surface Fitting Method for 3D Reconstruction from 2D Video Sequence

The objective of Task 9 is to design an automatic surface fitting method for 3D reconstruction from 2D video sequence.

3D reconstruction is one of the most fundamental problem in computer vision and computer graphics. 3D video reconstruction is the process of recovering the 3D geometric structure and surface from a 2D video sequence which is one of the most challenging research topics in 3D reconstruction. The challenge in 3D video reconstruction is how to align 2D image sequence pixel by pixel. Traditional stereo reconstruction methods and volumetric reconstruction methods suffer from the blank wall problem and the estimated dense depth map is not smooth for surface modeling. In this paper, We present a novel surface fitting approach for 3D dense reconstruction. We propose a non-linear deterministic annealing algorithm to decompose the 3D sparse structure to separate regions, and estimate the dense depth map by plane surface fitting. The experimental results show that the new approach can segment the 3D space geometrically and generate smoother dense depth map.

The results of Task 9 were reported in Ref. [109]. Next, we present the technical details.

11.1 Introduction

3D reconstruction is one of the most challenging and fundamental problem in the area of computer vision. During the recent years, a lot of approaches were developed for modeling and rendering the virtual scene from 2D videos and image sequences [110] [111] [112] [113]. Currently, most of the systems and applications in 3D reconstruction are used for visual inspection and architecture modeling. However, there is more demand for 3D entertainment, for example, 3D movies. The change of demand results in an attention for smooth visual quality of the reconstructed scene. In this case, visual quality of the virtual scene becomes the dominant factor. While the foremost goal in previous approaches is the accuracy of the position of each point in 3D geometry.

In the last two decades, tremendous progress has been made on self-calibration and 3D surface modeling [114] [115] [116] [117]. Most of the methods use 2D video sequences or 2D images as input and try to retrieve the depth information of the scene captured by the input video sequence. The estimated depth information helps to reconstruct the full 3D view of the scene. The existing techniques are able to well calculate the camera motion and compute a sparse depth map from the original image sequence [118] [119] [120] [110] [121]. However, fully reconstruction of a 3D scene requires the depth information of much more image pixels which requires the alignment of almost all pixels of the input images. This problem is known as dense matching problem [122] [123] [124].

A traditional solution to the dense matching problem is called epi-line searching. Epi-line search method uses the geometric constraints to degrade a 2D searching to a 1D range searching [125] [126] [127]. Although the search is constraint to 1D which seems easier to search, the blank wall problem, which is not solved in 2D feature correspondence, still exist in epi-line search. The blank wall problem is that given a texture less blank wall, it is very hard to find an accurate pixel to pixel correspondence across the input images.

Another solution to the dense matching problem is volumetric reconstruction method. Lhuillier and Quan proposed a quasi-dense approach to surface reconstruction in which they used a best first search based on combined 3D and 2D information [112] [128]. Instead of using pixel-based searching and matching, volumetric reconstruction takes the scene as a tessellation of 3D cubes, called voxels. Each voxel may be either empty or occupied by the scene structure. Various methods has been proposed to build the volumetric model which is used to generate the most consistent projections with the original images. Volumetric reconstruction could well recover the scene of the moving foreground, however, it is hard to reveal the static background structure using volumetric methods.

In this section, we propose a novel 3D dense reconstruction method based on geometric segmentation and surface fitting. We use the existing techniques for feature correspondence, projective reconstruction and self-calibration to get the sparse points reconstruction. To address the dense matching problem, we use geometric segmentation to segment the 3D space into several separate regions, and for each region, we estimate the dense 3D depth map by surface fitting. We propose a non-linear deterministic annealing algorithm in order to partition the 3D space geometrically. With the assumption that each subspace could be modeled by a linear plane, we can retrieve the depth information for each pixel using surface fitting. The new approach is able to generate a much smoother 3D dense reconstruction comparing to the traditional methods.

This section is organized as follows. Section 11.2 present the background and problem formulation. We present the system scheme for 3D reconstruction in Section 12.2. Then we solve the geometric segmentation and surface fitting problem in Section 11.4. The experimental results are shown in Section 12.5.

11.2 Background and Problem Formation

In this section, we briefly review the 3D reconstruction techniques and formulate the geometric fitting problem mathematically.

11.2.1 3D Reconstruction

3D reconstruction has been one of the most fundamental research topics in computer vision for decades. Although they may differ in some specific part, most 3D reconstruction approaches are generally based on the same pipeline [127]. The pipeline is given in Fig. 73.

The first step in 3D reconstruction from a video sequence is to group the whole video sequence into several scenes by key frames. For each scene, motion detection is needed to find moving regions from the static background. In the later part, moving foreground and static background will be treated separately and then combined together to reconstruct the scene as a whole.

The second step is sparse reconstruction. Sparse reconstruction includes several component, feature correspondence, projection reconstruction and Euclidean reconstruction. The camera motion is estimated and The Euclidean structure of the static background scene is recovered. For the moving regions, we introduce the virtual camera concept and apply the same reconstruction algorithm to recover the 3D structure. During the last two decades, tremendous progress has been made to camera self-calibration and structure computation. Sparse reconstruction starts from feature correspondence which is the most crucial part of the process. The goal of Image correspondence, also called feature correspondence, is to align different images, from a video sequence or taken separately, by finding corresponding points that describe the same point in 3D geometry [129] [130]. As known to all, not all points are suitable for matching or tracking through different images, so only a few points are selected as feature points for matching [131]. So sparse reconstruction only rely on a number of distinct points which is different from the following dense reconstruction which require the correspondence of all points, if possible. Furthermore, feature points may be mismatched, known as outliers [101], which may restrict the accuracy of the reconstruction result. Given correctly matched feature points from two input images, projection reconstruction is to find the relative pose between the two views. The projective structure is mathematically expressed by fundamental matrix. Given sufficient corresponding feature points, with the assumption that the world frame is the same frame as that of the first image, we are able to compute the fundamental matrix. The projective reconstruction is determined by an arbitrary projective transformation. To solve this problem, canonical decomposition is applied to fix a particular choice of projective transformation. Therefore, the projective structure is not suitable for visualization and an update

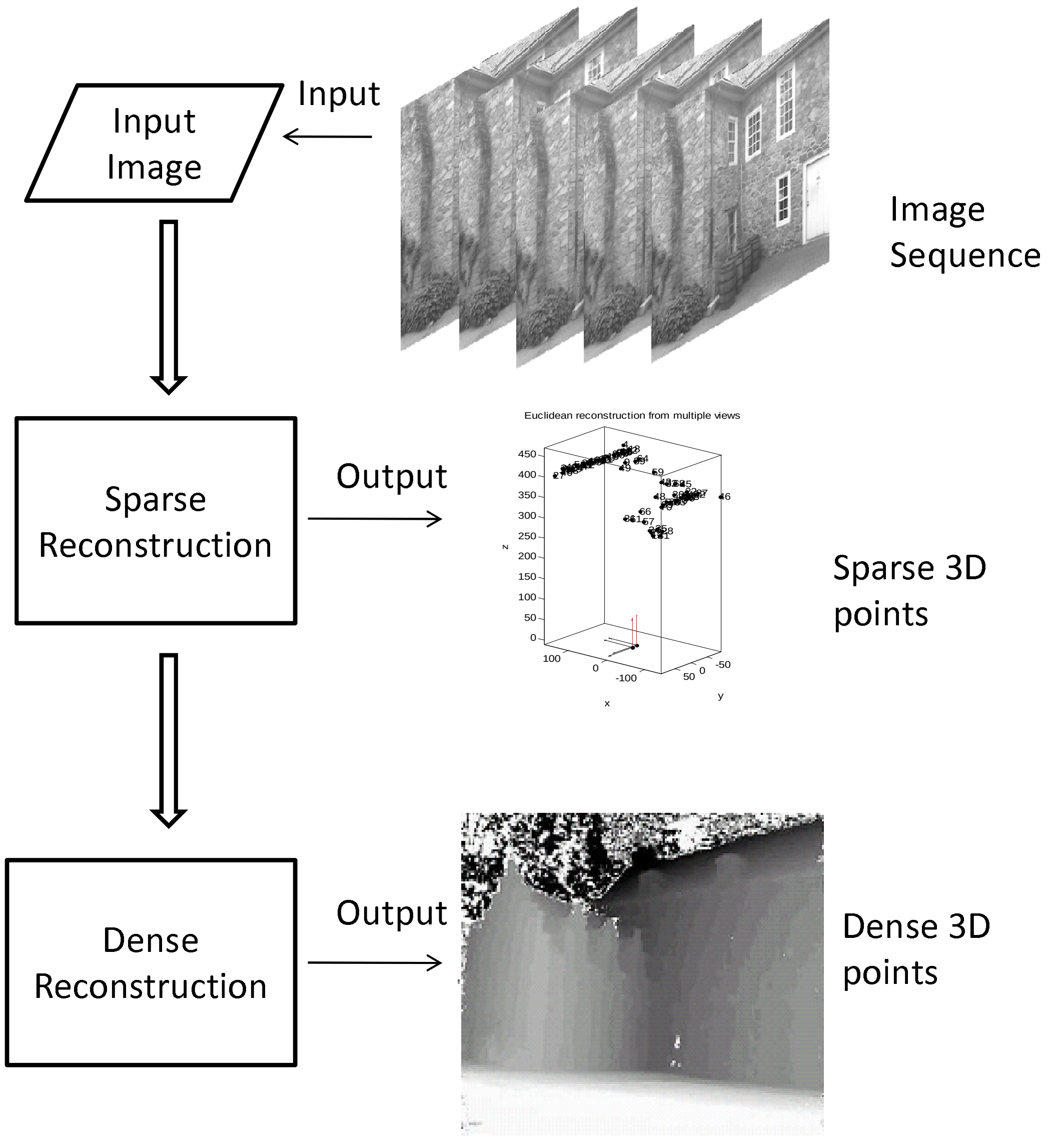


Figure 64: The pipeline for 3D video reconstruction system.

to a full-fledged Euclidean reconstruction is required to recover the metric 3D geometric structure. The update to a metric structure, determined up to an unknown scalar factor, needs the information of intrinsic parameters of the camera. Since we have no prior knowledge of the camera, this approach is called self-calibration and has received a lot of attention in recent years. The approach we present here is called absolute conic constraint, or absolute quadric constraints.

The sparse reconstruction gives a sparse structure of the desired scene; however, it could not give a satisfied visual presentation. Thus, we still need to compute the depth of a lot more points, which is known as dense reconstruction or surface reconstruction. The traditional approaches for dense reconstruction could be classified as two approaches, namely stereoscopic reconstruction and volumetric reconstruction. In this section, we propose a novel approach to obtain the static background structure. Unlike the previous approach, we apply geometrical segmentation and surface fitting instead of dense searching and matching. Here we assume that the static background could be decomposed of several uniform regions or regular surfaces. We can then segment the whole surface into several regions based on their geometric properties. For each region, we obtain a mathematical expression by surface fitting. With the assumption that each region has sufficient number of sparse feature points, combined with the sparse depth map, we could then compute the depth information by fitting each pixel within the estimated surface. Combining the depth map of different regions, we could finally obtain the depth map of the whole scene. The merit of this approach is that it well handles uniform regions and occlusions by mismatching issues. Also, the result is smoother than traditional stereoscopic reconstruction algorithms. The geometric fitting problem is formulated in subsection 11.2.2 and we give the solution to the problem in details in Section 11.4.

11.2.2 Geometric Fitting

The classic geometric fitting problem is to find a geometrical surface that best fits to a set of 3D points. Geometric fitting is commonly used in 3D model fitting and 3D visual reconstruction in computer vision.

Given a 3D point data set $\mathfrak{X} = \{\mathbf{x}_i\}, \mathbf{x}_i \in R^3, i = 1, 2, ..., n$, the geometrical fitting problem is usually stated as the optimization of a cost that measures how the geometrical surface function $\mathfrak{G} = \{\mathbf{x} : g_\theta(\mathbf{x}) = 0\}$ fits the data set $\mathfrak{X}$. The most commonly used objective function is the least squares cost,

$$D = \sum_{i=1,\ldots N} d(\mathbf{x}_i, g_\theta)^2 \tag{127}$$

$$d(\mathbf{x}_i, g_\theta) = \min \|\mathbf{x}_i - \mathbf{x}_j\|^2, \;\; \mathbf{x}_j \in \mathfrak{S} \tag{128}$$

The fitting function g_θ is learned by minimizing the design cost, D, measured over the input data set, $\mathfrak{X}$. It is well-known that for most choices of D, the cost measured during design monotonically decreases as the size of the learned fitting function g_θ is increased. With a large set of functions, it is easy to create a surface which passes through each input data point but is suspiciously complicated. The principle of Occam's razor states that the simplest model that accurately represents the data is most desirable. So we prefer to use a few basis functions which yield a smoother, simpler surface which could well approximates the original data. Generally, there are two approaches to solve the over fitting problem. One approach is to add penalty terms to the data set, like smoothness or regularization constraints. Another approach is to first build a large model and then remove some parameters by retaining only the vital model structure. Although both approaches can generate parsimonious models, the descent based learning methods all suffer from a serious limitation. The non-global optima of the cost surface may easily result in poor local minima to the descent based learning methods. Techniques adding penalty terms to the cost function further increases the complexity of the cost surface and worsen the local minimum problem.

One of the most popular clustering algorithm is Lloyd's algorithm, which starts by partitioning the input data into k initial sets. It calculates the centroid of each set via some metric. Lloyd's algorithm iteratively associates each point with the closest centroid and recalculates the centroids of the new clusters. Although widely used in real world applications, there are two serious limitations of Lloyd's algorithm. The first limitation is that the partitioning result depends on the initialization of the cluster centers, which may lead to poor local minima. The second limitation is that Lloyd's algorithm can only partition linear separable clusters. In order to avoid initialization dependence, a simple but useful solution is to use multiple restarts with different initializations to achieve a better local minima. Global k-means [103] is proposed to build the clusters deterministically, which use the original k-means algorithm as a local search step. At each step, global k-means add one more cluster based on previous partitioning result. Deterministic annealing [104] is another optimization technique to find a global minimum of a cost function. Deterministic annealing explore a larger cost surface by introducing a constraint of randomness. At each iteration, the randomness is constrained and a local optimization is performed. Finally, the imposed randomness is reduce to zero, and the algorithm optimizes over the original cost function. Kernel method [105] is used to solve the second problem by mapping the data points from input space to a higher dimensional feature space through a non-linear transformation. Then the optimization is applied in the feature space. The linear separation in the feature space turns out to be a non-linear separation in the original input space.

11.3 3D Video Reconstruction

Here, we simply introduce the 3D reconstruction algorithm proposed in Ma et. al's book [116] on which our experiments are based. When developing a stereo vision algorithm for registration, the requirements for accuracy vary from those of standard stereo algorithms used for 3D reconstruction. For example, a multi-pixel disparity error in an area of low texture, such as a white wall, will result in significantly less intensity error in the registered image than the same disparity error in a highly textured area. In particular, edges and straight lines in the scene need to be rendered correctly.

11.3.1 Overview of 3D Reconstruction System

The 3D reconstruction algorithm is implemented in the following steps. First, geometric features are detected automatically in each individual images. Secondly, feature correspondence is established across all the images. Then the camera motion is retrieved and the camera is calibrated. The Euclidean structure of the scene is recovered afterward. After that, we apply the geometric segmentation algorithm described in Section 11.4. Finally the dense depth map is reconstructed by geometric fitting. The system scheme is given in Fig. 65.

11.3.2 Feature Selection

The first step in 3D reconstruction is to select candidate features in all images for tracking across different views. Ma et al. [116] use point feature in reconstruction which is measured by Harris' criterion,

$$C(\mathbf{x}) = \det(G) + k \times \text{trace}^2(G) \tag{129}$$

where $\mathbf{x} = [x, y]^T$ is a candidate feature, $C(\mathbf{x})$ is the quality of the feature, k is a pre-chosen constant parameter and G is a 2×2 matrix that depends on $\mathbf{x}$, given by

$$G = \begin{bmatrix} \sum_{W(\mathbf{x})} I_x^2 & \sum_{W(\mathbf{x})} I_x I_y \\ \sum_{W(\mathbf{x})} I_x I_y & \sum_{W(\mathbf{x})} I_y^2 \end{bmatrix} \tag{130}$$

where $W(\mathbf{x})$ is a rectangular window centered at $\mathbf{x}$ and I_x and I_y are the gradients along the x and y directions which can be obtained by convolving the image I with the derivatives of a pair of Gaussian filters. The size of the window can be decided by the user, for example 7×7. If $C(\mathbf{x})$ exceeds a certain threshold, then the point $\mathbf{x}$ is selected as a candidate point feature.

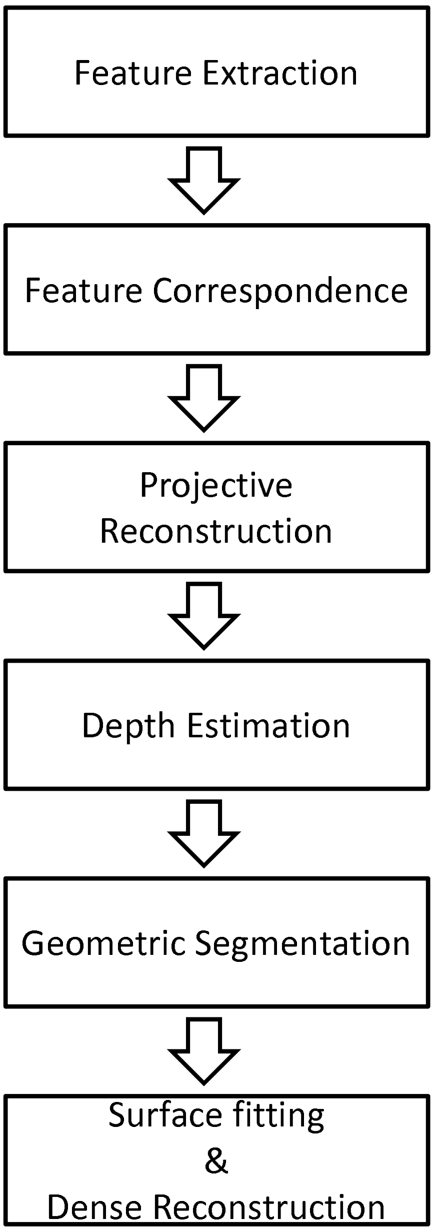


Figure 65: The scheme for 3D video reconstruction system.

11.3.3 Feature Correspondence

Once the candidate point features are selected, the next step is to match them across all the images. In this subsection, we use a simple feature tracking algorithm based on a translational model.

We use the sum of squared differences (SSD) as the measurement of the similarity of two point features. Then the correspondence problem becomes looking for the displacement **d** that satisfies the following optimization problem:

$$\min_{\mathbf{d}} \sum_{\mathbf{x} \in W(\mathbf{x})} [I_2(\mathbf{x}+\mathbf{d}) - I_1(\mathbf{x})]^2 \tag{131}$$

where $\mathbf{d}$ is the displacement of a point feature of coordinates $\mathbf{x}$ between two consecutive frames I_1 and I_2. Lucas and Kanade also give the close form solution of 169

$$\mathbf{d} = -G^{-1}\mathbf{b} \tag{132}$$

where

$$\mathbf{b} \doteq \begin{bmatrix} \sum_{W_{(\mathbf{x})}} I_x I_t \\ \sum_{W_{(\mathbf{x})}} I_y I_t \end{bmatrix} \tag{133}$$

G is the same matrix we used to compute the quality of the candidate point feature in Eq. 167, and $I_t \doteq I_2 - I_1$.

11.3.4 Estimation of Camera Motion Parameters

In this subsection, we recover the projective structure of the scene from the established feature correspondence. We will follow the notation used in Ma et al.'s book [116]. For the detail of the proof of this algorithm, please refer to the reference.

The reconstruction algorithm is based on a perspective projection model with a pinhole camera. Suppose we have a generic point $p \in \mathbb{E}^3$ with coordinates $\mathbf{X} = [X, Y, Z, 1]^T$ relative to a world coordinate frame. Given two frames of one scene which is related by a motion $g = (R, T)$, the two image projection point $\mathbf{x}_1$ and $\mathbf{x}_2$ are related as follows:

$$\lambda_1 \mathbf{x}'_1 = \Pi_1 \mathbf{X}_p, \quad \lambda_2 \mathbf{x}'_2 = \Pi_2 \mathbf{X}_p \tag{134}$$

where $\mathbf{x}' = [x, y, 1]^T$ is measured in pixels, λ_1 and λ_2 are the depth scale of $\mathbf{x}_1$ and $\mathbf{x}_2$, $\Pi_1 = [K, 0]$ and $\Pi_2 = [KR, KT]$ are the camera projection matrices and K is the camera calibration matrix. In order to estimate λ_1, λ_2, Π_1 and Π_2, we need to introduce the epipolar constraint. From Equation (172), we have

$$\mathbf{x}'^T_2 K^{-T}\hat{T}RK^{-1}\mathbf{x}'_1 = 0 \tag{135}$$

The fundamental matrix is defined as:

$$F \doteq K^{-T}\hat{T}RK^{-1} \tag{136}$$

With the above model, we could estimate the fundamental matrix F via the Eight-point algorithm. Then we could decompose the fundamental matrix to recover the projection matrices Π_1 and Π_2 and the 3D structure. We only give the solution here by canonical decomposition:

$$\Pi_1 p = [I, 0], \Pi_2 p = [(\widehat{T'})^T F, T'], \lambda_1\mathbf{x}'_1 = \mathbf{X}_p, \lambda_2\mathbf{x}'_2 = (\widehat{T'})^T F\mathbf{X}_p + T' \tag{137}$$

11.3.5 Depth Estimation

The Euclidean structure $\mathbf{X}_e$ is related to the projective reconstruction $\mathbf{X}_p$ by a linear transform $H \in \mathbb{R}^{4\times4}$,

$$\Pi_{ip} \sim \Pi_{ie}H^{-1}, \mathbf{X}_p \sim H\mathbf{X}_e, i = 1, 2, ..., m \tag{138}$$

where $\sim$ means equality up to a scale factor and

$$H = \begin{bmatrix} K & 0 \\ -\nu^T K & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \tag{139}$$

With the assumption that K is constant, we could estimate the unknowns K and ν with a gradient decent optimization algorithm. In order to obtain a unique solution, we also assume that the scene is generic and the camera motion is rich enough.

11.3.6 Geometric Segmentation

As we have discussed before, not all points in an image are suitable for matching or tracking. The feature points that we have selected are only a bunch of distinct points. Therefore, the previous reconstruction is a sparse scene reconstruction. The sparse structure is not suitable for human visualization. For this reason, a dense matching is necessary to establish a 3D geometric view.

In this section, we propose a new dense matching method based on geometric segmentation. We first segment the surface of the 3D scene into several regions based on the geometric relationship. For each small homogeneous surface, we are able to model it by a plane. With the depth information of the feature points that we already get from the sparse reconstruction, we could compute the depth information for each pixel in the entire region. Since the depth information we obtained is based on a plane model, the image rendered from the 3D model is much smoother than the traditional approaches. In order to simplify the problem of surface fitting, we first segment the input image based on its geometric structure. It is different from the traditional object based image segmentation. The segmentation process is critical because proper segmentation could simplify the surface fitting. On the contrary, improper segmentation which combines too many surface areas will increase the complexity of surface modeling.

Due to the fact that the 3D data is localized to a few relatively dense clusters, we design a non-linear function to map the data point from geometrical space to surface model space and apply deterministic annealing in the feature space to partition the feature space into several regions with different sizes and shapes. For each region, we can easily find a linear plane model to fit the data. Non-linear deterministic annealing method offers three important features: 1) the ability to avoid many poor local optima; 2) the ability to minimize the cost function even its gradients vanish almost everywhere; 3) the ability to achieve non-linear separation. However, there is no close form solution for non-linear deterministic annealing problem, therefore we use a gradient descent algorithm to solve this problem. The details of this algorithm is discussed in Section 11.4.

11.3.7 Depth Recovery

Here, we only consider two images. Suppose for the first image, we have the 3D point set $\mathbf{X}_e^j, j = 1, 2, ..., n$ which could be divided into three clusters,$\mathbf{X}_{e1}$, $\mathbf{X}_{e2}$, $\mathbf{X}_{e3}$. For each cluster, there are at least three non-collinear points. Then we could have the plane model for this cluster. Let's take the example of $\mathbf{X}_{e1}$, suppose there are m points in the cluster and we have the plane model as follows:

$$\mathbf{A} \cdot p = 1 \tag{140}$$

where $\mathbf{A} = [\mathbf{X}_{e1}^i], i = 1, ..., m$ and $p = [a, b, c]^T$ is the plane parameter.

Given an arbitrary point $\mathbf{x}^i = [x^i, y^i]^T$ measured in pixels in the first cluster, we could estimate it's depth scale λ^i by solving the following equation.

$$\lambda^i \mathbf{x}'^i = H_1^{-1} \Pi_1 \mathbf{X}_e^i \tag{141}$$

where $\mathbf{x}'^i = [x^i, y^i, 1]^T$, H_1^{-1} and Π_1 are estimated in previous subsections. In Eq. 196, only λ^i is unknown and with the constraint on $\mathbf{X}_e^i$ with Eq. 195, we could easily get the value of λ^i.

Then, with $\Pi_1 = [I, 0]$, we could have $X_p^i = [\lambda_1^i x^i, \lambda_1^i y^i, \lambda_1^i, 1]$. from Eq. 172, we can get the relation between two image projection point $\mathbf{x}_1^i$ and $\mathbf{x}_2^i$ as follows:

$$\widehat{\mathbf{x}_2^{\mathbf{i}}}' = \Pi_2 \mathbf{X}_p^i \tag{142}$$

where $\widehat{\mathbf{x}_2^{\mathbf{i}}}' = [\lambda_2^i x_2^i, \lambda_2^i y_2^i, \lambda_2^i]$. We could then get the position of the corresponding point $\mathbf{x}_2^i = [x_2^i, y_2^i]$ in the second image.

11.4 Geometric Segmentation based Dense Reconstruction

As we have discussed, not all points in an image are suitable for matching or tracking. The feature points that we have selected are only a bunch of distinct points. Therefore, the first reconstruction is a sparse reconstruction. The sparse structure is not suitable for human visualization. For this reason, a dense matching is necessary to establish a 3D geometric view. As known to all, the most popular solution for dense matching is based on the epi-polar constraint. This approach uses geometric constraints to restrict correspondence search from 2D to 1D range. The main disadvantages of this approach are that the dense depth map is not smooth because of outliers. Lhuillier and Quan proposed another dense matching method called quasi-dense approach. They tried to combine 3D data points and 2D image information. However, the visual problem still exists.

In this section, we propose a non-linear deterministic annealing approach for space partitioning in 3D Euclidean space. We use deterministic annealing to divide the input space into several regions with different sizes and shapes. With the partition, we can easily find a linear local surface to fit the data within each region. Deterministic annealing method offers two great features: 1) the ability to avoid many poor local optima; 2) the ability to minimize the cost function even its gradients vanish almost everywhere. Due to the fact that the data is localized to a few relatively dense clusters, we design a non-linear function to map the data point from the geometric space to surface feature space and apply deterministic annealing in the feature space instead of the geometric space. The advantage of our approach is that the estimated dense depth map is much more smooth than the traditional approaches.

Given a set of data $\mathfrak{X}$ of scattered 3D points, we would like to find the geometric surface that best fits to the scattered data. The fitting problem is usually stated as the optimization of a cost that measures how well the fitting function $g(\mathbf{x}_i)$ fits the data. The most commonly used objective function is the least squares cost. Finding a good fit is a challenging problem and may be more of an art than a science. If we use a large set of functions as the basis, we may create a surface which passes through each data point but is suspiciously complicated. Using few basis functions may yield a smoother, simpler surface which only approximates the original data. Due to the over fitting problem, we propose an new approach to optimize the objective function via space partitioning. We first partition the data set into several subsets such that the data points $\mathbf{x}$ in each subset could be approximated by a linear surface model. In other words, we would like to use a set of plain models to approximate the date set. The objective of space partitioning is to minimize the geometric fitting error.

$$\min_{g_{\theta_k}} \sum_{k=1}^{K} \sum_{i \in C_k} d(\mathbf{x}_i, g_{\theta_k}) \tag{143}$$

where, $\mathbf{x}_i = [x_i, y_i, z_i]^T$ is the i-th point data, $\theta_k = [a_k, b_k, c_k]^T$ is the k-th linear surface model, and $d_{i,k}$ is is the fitting error between $\mathbf{x}_i$ and plane model $g_{\theta_k} = 0$ which is defined as

$$d_{i,k} = d(\mathbf{x}_i, g_{\theta_k}) = \frac{(\mathbf{x}_i^T g_{\theta_k} - 1)^2}{a_k^2 + b_k^2 + c_k^2} \tag{144}$$

11.4.1 Deterministic Annealing

The deterministic annealing (DA) approach [104] to clustering has demonstrated substantial performance improvement over traditional supervised and unsupervised learning algorithms. DA mimics the annealing process in static The advantage of deterministic annealing is its ability to avoid many poor local optima. The reason is that deterministic annealing minimizes the designed cost function subject to a constraint on the randomness of the solution. The constraint, Shannon entropy, is gradually lowered and eventually deterministic annealing optimize on the original cost function. Deterministic annealing mimics the simulated annealing [106] in statistical physics by the use of expectation. Deterministic annealing derives an effective energy function through expectation and is deterministically optimized at successively reduced temperatures. The deterministic annealing approach has been adopted in a variety of research fields, such as graph-theoretic optimization and computer vision. A. Rao et al. [107] extended the work for piecewise regression modeling. In this subsection, we will briefly review their work.

Given a data set $(\mathbf{x}, \mathbf{y})$, the regression problem is to optimize the cost that measures how well the regression function $f(\mathbf{x})$ approximates the output $\mathbf{y}$, where $\mathbf{x} \in \mathcal{R}^m$, $\mathbf{y} \in \mathcal{R}^n$, and $g : \mathcal{R}^m \rightarrow \mathcal{R}^n$. In the basic space partitioning approach, the input space is partitioned into K regions and the cost function becomes

$$\min_{g_{\theta_k}} \sum_{k=1}^{K} \sum_{i \in C_k} d(\mathbf{y}_i, f(\mathbf{x}_i, g_{\theta_k}) \tag{145}$$

where $d(\cdot,\cdot)$ is the distortion measure function. Instead of seeking the optimal hard partition directly, randomness is introduced for randomized assignment for input samples.

$$D = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) d(\mathbf{y}_i, f(\mathbf{x}_i, g_{\theta_k}) \tag{146}$$

In A. Rao et al.'s work, they use the nearest prototype (NP) structure as constraint and given the set of prototypes$\{\mathbf{s}_j : j = 1, 2, 3, ..., K\}$ in the input space, a Voronoi criterion is defined for NP partition

$$C = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) ||\mathbf{x}_i - \mathbf{s}_j|| \tag{147}$$

Although the ultimate goal is to find the hard partition, some "randomness" is desired during the assignment. Shannon entropy is introduced as a constraint of the randomness.

$$H = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} P(\mathbf{x}_i \in C_j) \log P(\mathbf{x}_i \in C_j) \tag{148}$$

Eventually, this constrained optimization problem could be rewritten as the minimization of the corresponding Lagrangian

$$\min_{\{\mathbf{\Lambda_j}\}\{\mathbf{s_j}\},\gamma} \quad F = D - TH \tag{149}$$

where, γ is a nonnegative Lagrange multiplier which controls the randomness of the space partition.

11.4.2 Non-linear Deterministic Annealing

In this section, we propose a new approach based on non-linear deterministic annealing to solve the 3D geometric fitting problem. We first use a non-linear function to map the input point data to a high dimensional feature space using the local geometric structure of the data. Then we apply deterministic annealing in the feature space to leverage the local geometric structure for clustering.

To solve the space partitioning problem, we do not use prototype to calculate the difference. The reason is that the prototype in space partitioning is generally not sufficient to represent a plane in 3D space. Instead, we estimate the linear plane model and calculate the fitting error as the Euclidean distance between the data and the plane. The traditional local optimization algorithm will likely stuck at a local optima. In order to avoid local optima, we use local geometric structure from neighboring data points and embedded the data vectors to a higher dimension as follows.

The input data is given as a 3D point, $\mathbf{x}_i = [x_i, y_i, z_i]^T$. With the assumption that nearest data points are on the same plane, we could estimate the local plane model, $\mathbf{L}_i = [a_i, b_i, c_i]^T$ of data point $\mathbf{x}_i$ and its K nearest neighbor points.

$$\mathbf{L} = \begin{bmatrix} a(\mathfrak{X}) \\ b(\mathfrak{X}) \\ c(\mathfrak{X}) \end{bmatrix} \tag{150}$$

$$\mathbf{f} = \begin{bmatrix} \mathbf{x} \\ \mathbf{L} \end{bmatrix} \tag{151}$$

Then we revise the distortion function as follows,

$$D(\mathbf{f}_i, g_{\theta_j}) = D_1(I_1\mathbf{f}_i, g_{\theta_j}) + D_2(I_2\mathbf{f}_i, g_{\theta_j}) \tag{152}$$

$$I_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{153}$$

$$I_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{154}$$

where $D_1 = d_{i,j}$ calculate the fitting error between the data point and the estimated plane, and D_2 calculate the difference between the local estimated plane model and the cluster scale estimated plane model. D_2 is defined as follows:

$$D_2(I_2\mathbf{f}_i, g_{\theta_j}) = \frac{I_2\mathbf{f}_i^T \times g_{\theta_j}}{|I_2\mathbf{f}_i| \times |g_{\theta_j}|} \tag{155}$$

After the mapping, we apply deterministic annealing algorithm to partition the data into several clusters as follows.

$$\min_{g_{\theta_j}} F = D - TH \tag{156}$$

where $g_{\theta_j} = [a_j, b_j, c_j]$ is the geometrical surface model parameter to be estimated, D is the sum of square of geometrical fitting error and H is the entropy constraint. We define D and H as follows:

$$D = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j}) d(\mathbf{x}_i, g_{\theta_j}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i) d(\mathbf{x}_i, g_{\theta_j}) \tag{157}$$

$$H(\mathbf{X}, g_\theta) = \sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j}) \log p(\mathbf{x}_i, g_{\theta_j}) \tag{158}$$

To perform optimization we need to further analyze its terms. We can rewrite equation (186) by applying the chain rule of entropy as

$$H(\mathbf{X}, g_\theta) = H(\mathbf{X}) + H(g_\theta|\mathbf{X}) \tag{159}$$

Notice that the first term $H(\mathbf{X})$ is the entropy of the source and is therefore constant with respect to the cluster g_{θ_j} and association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$. Thus we can just focus on the conditional entropy

$$H(g_\theta|\mathbf{X}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i) \log p(g_{\theta_j}|\mathbf{x}_i) \tag{160}$$

The minimization of F with respect to association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$ gives rise to the Gibbs distribution

$$p(g_{\theta_j}|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T})}{Z_x} \tag{161}$$

where the normalization is

$$Z_x = \sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T}) \tag{162}$$

The corresponding minimum of F is obtained by plugging equation (189) back into equation (184)

$$F^* = \min_{p(g_{\theta_j}|\mathbf{x}_i)} F = -T \sum_{i=1}^{N} p(\mathbf{x}_i) \log Z_x \tag{163}$$

To minimize the Lagrangian with respect to the cluster model g_{θ_j}, its gradients are set to zero yielding the condition

$$\nabla_{g_{\theta_j}} F = \frac{1}{N} \sum_{i=1}^{N} p(g_{\theta_j}|\mathbf{x}_i) \nabla_{g_{\theta_j}} d(\mathbf{x}_i, g_{\theta_j}) = 0 \tag{164}$$

Since there is no close form solution for non-linear deterministic annealing problem, we use a gradient descent algorithm to solve this problem. I present our algorithm in Figure. 78.

11.5 Experimental Results

In this section, I first compared three geometric segmentation algorithms, Projection based iterative geometric segmentation algorithm (PI), Adaptive projection based iterative algorithm (API), and non-linear DA based geometric segmentation algorithm(NDA), based on both synthetic data and real world data.

11.5.1 NDA on Synthetic Data

The purpose of the first experiment is to compare NDA, PI, and API on synthetic data with ground truth. I generated the synthetic data using MATLAB 'randperm' function. The data is a set of 3D points on several linear planes without noise. In this experiment, I run each algorithm for 1000 times. Each time, a random data set is generated and used. We segment the same data set with different algorithms and calculate the average squared approximation error. Below is the experimental result in Table. 12. K represents the number of planes in a test data set. For each plane, 100 random points are generated. The date set 1 contains 300 data in total from 3 non parallel planes. The data set 2 contains 400 data from 4 planes. The data set 3 contains 500 data from 5 planes and the data set 4 contains 600 data from 6 planes. The average squared approximation error of NDA is ignorable comparing to the errors of PI and NPI. From the experimental result, we can say that NDA algorithm outperforms both PI and API algorithms in the average squared approximation error. The reason NDA algorithm outperforms PI and API algorithms is that NDA is able to separate the space non-linearly and avoid many poor local optima.

We also measure the performance of the segmentation algorithms in percentage of correct identification of planes. We test the same data set as used in the previous experiment and compute the correct identification percentage averaging over all tests. Below is the experimental result in Table. 13. We observed that correct identification rates of NDA and API are much higher than the

```
1.  Algorithm 78 **KDA based geometrical segmentation algorithm**
2.  **Set Limit**
3.      $K_{max}$: maximum number of clusters
4.      $T_{init}$: starting temperature
5.      $T_{min}$: minimum temperature
6.      $\delta$: perturbation vector
7.      $\alpha$: cooling rate (must be $< 1$)
8.      $I_{max}$: maximum iteration number
9.      $th$: Iteration threshold
10.     $sth$: Surface distance threshold
11. **Initialization**
12.     $T = T_{init}, K = 2, \Lambda_1 = (X^T X)^{-1} X^T \vec{1}, \Lambda_2 = \Lambda_1, [p(\mathbf{\Lambda}_1|\mathbf{x}_i), p(\mathbf{\Lambda}_2|\mathbf{x}_i)] = [\frac{1}{2}, \frac{1}{2}], \forall i.$
13. **Perturb**
14.     $\Lambda_j = \Lambda_j + \delta, \forall j.$
15.     $L_{old} = D - TH.$
16. **Loop until convergence,** $i = 0 \; \forall j$
17.     For all $\mathbf{x}_i$ in the training data, compute the association probabilities
            $$p(\mathbf{\Lambda}_j|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i, \mathbf{\Lambda}_j)}{T})}{\sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i, \mathbf{\Lambda}_j)}{T})} \quad (165)$$
18.     update the surface model
            $$\mathbf{\Lambda}_j \longleftarrow \mathbf{\Lambda}_j + \alpha \nabla_{\mathbf{\Lambda}_j} F. \quad (166)$$
19.     i = i+1;
20.     if ($i > I_{max}$ or $\nabla_{\mathbf{\Lambda}_j} F < th$ ) End **Loop**
21. **Model Size Determination**
22.     if($d(\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}) < sth$)
23.         replace $\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}$ by a single plane
24.     $K$ =number of planes after merging
25. **Cooling Step**
26.     $T = \alpha T.$
27.     if ($T < T_{min}$)
28.         perform last iteration for $T = 0$ and STOP
29. **Duplication**
30.     Replace each plane by two planes at the same location, $K = 2K$.
31. **Goto Step 10**
```

Table 12: The average squared approximation error.

K	PI	API	NDA
3	3.77×10^{-1}	3.00×10^{-9}	1.17×10^{-12}
4	4.01×10^{-1}	9.81×10^{-8}	2.21×10^{-12}
5	2.43×10^{-1}	2.86×10^{-9}	3.06×10^{-12}
6	2.94×10^{-1}	8.801×10^{-9}	3.00×10^{-12}

Table 13: The correct identification rate.

K	PI	API	NDA
3	83%	96%	99%
4	79%	93%	99%
5	82%	94%	97%
6	78%	97%	98%

correct identification rate of PI algorithm. The reason API algorithm outperforms PI algorithm is that API algorithm does not depends on random initialization while the segmentation results of PI algorithm heavily depends on initialization. Still NDA performs best among the three algorithms in correct identification rate.

11.5.2 NDA on Real World Data

In the second experiment, we test the geometric segmentation algorithm on some real world data. We use the 3D structure data set from the ‘housing’ image sequence. The data set includes 72 data points recovered by 3D reconstruction of 2D registered feature points. Most of the data points fall on the walls of the house in the image and we would like to estimate the surface model of the walls by geometric fitting. Fig. 67 shows the input 3D data points on the 1st frame of the ‘housing’ image sequence. The goal is to segment the data points into three groups and each group represent a wall in the image. Fig. 68 shows the geometric segmentation result by NDA algorithm and Fig. 69 shows the geometrical segmentation result by PI algorithm. It is pretty clear that NDA algorithm partitions the input data set into three clusters and each cluster represents a wall in the image. PI algorithm fails to find the geometric model of the walls and the data points are mixed. The experimental result on real world data shows that NDA algorithm can well segment the data sets based on their geometric relationship and the 3D surface is accurately recovered.

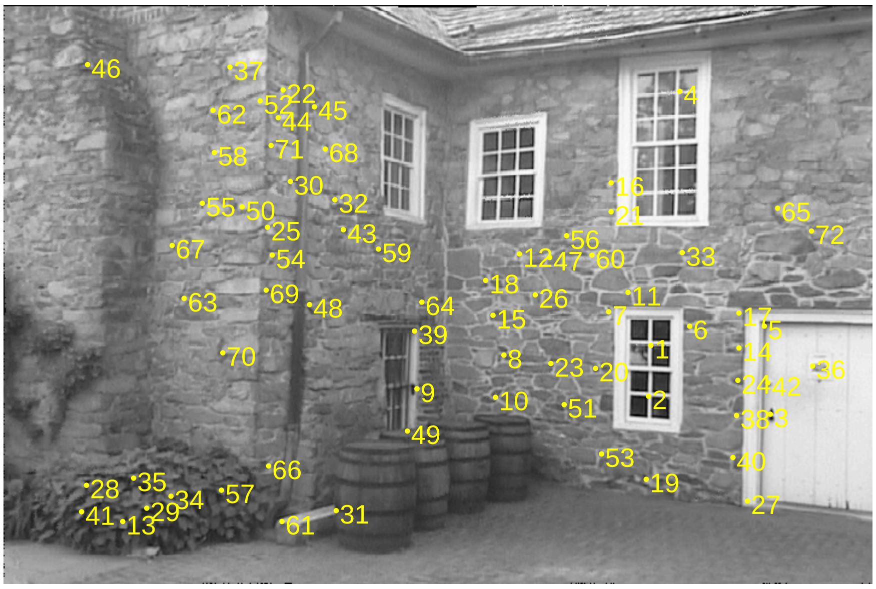


Figure 67: The input data points on the 1st frame of ‘housing’ image sequence.

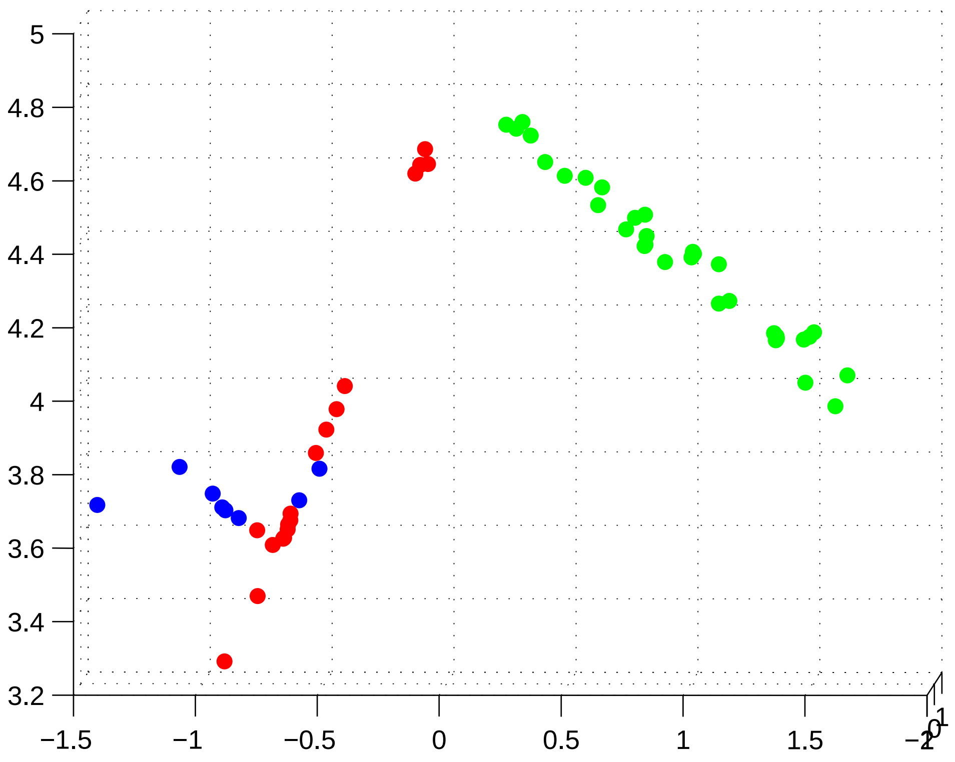


Figure 68: The geometrical segmentation result by the NDA algorithm of ‘housing’ data set.

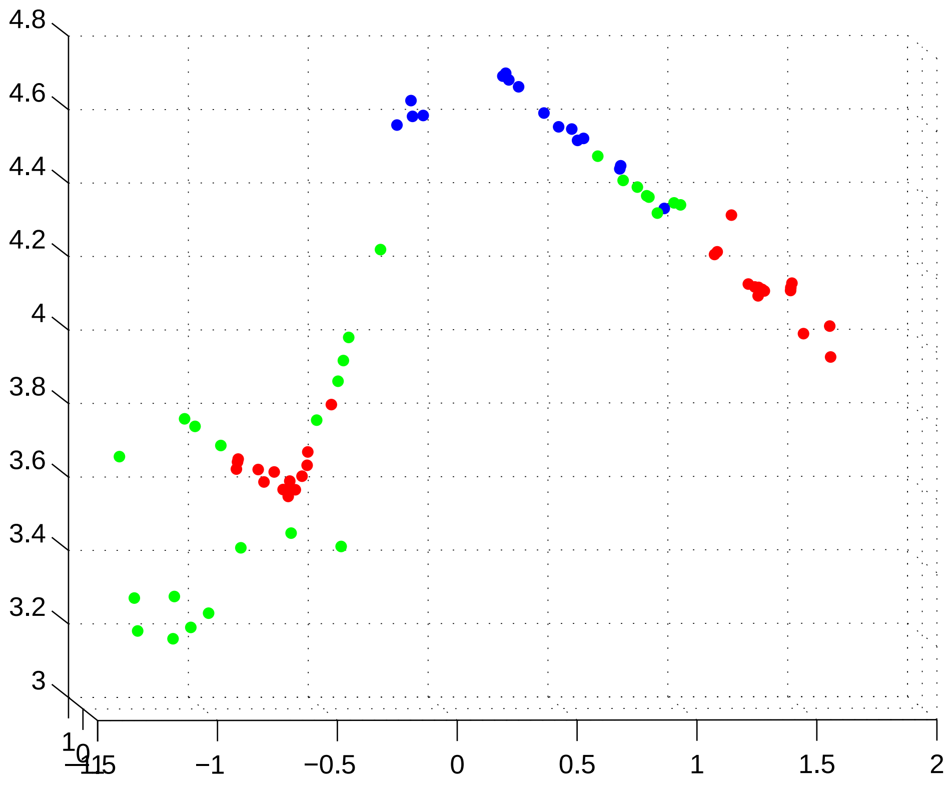
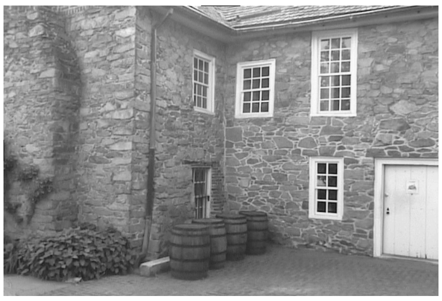


Figure 69: The geometrical segmentation result by the PI algorithm of ‘housing’ data set.

11.5.3 3D Video Dense Reconstruction

In the third experiment, we integrate the NDA algorithm in the 3D video reconstruction system. The input is an image sequence and the output is a dense depth map. In our experiment, we use the ‘oldhousing’ image sequence. Fig. 70 shows the first frame and the 88th frame of the test image sequence ‘oldhousing’. We first extract point features on all the input images. Then we apply feature correspondence algorithm to relate all the features. Fig. 67 show the selected feature points on the first frame. We then estimate the camera pose and intrinsic parameters. With the camera parameters, we are able to recover the sparse Euclidian structure of the feature points. Fig. 71 shows the estimated depth map of the selected feature points and the camera pose. After sparse reconstruction, we separate the 3D space into several regions using NDA algorithm. For each region, we use the surface fitting algorithm presented in Section 12.2 to estimate the depth information of each pixel. Combining the depth map of all regions, we can recover the 3D dense depth map of the whole frame. Fig. 72 shows the estimated dense depth map of the whole frame. Since we use surface fitting instead of searching for dense depth estimation, we do not need to worry about matching errors and outliers. The estimated dense depth map is very smooth and well represent the geometric structure of the 3D scene.


(a) The 1st frame in the ‘oldhousing’ video sequence

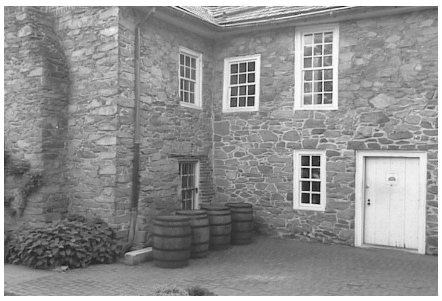

(b) The 88th frame in the ‘oldhousing’ video sequence

Figure 70: Original frames used for image registration

12 Depth Based Image Registration via Geometric Segmentation

The objective of Task 10 is to develop a depth-based image registration method via geometric segmentation.

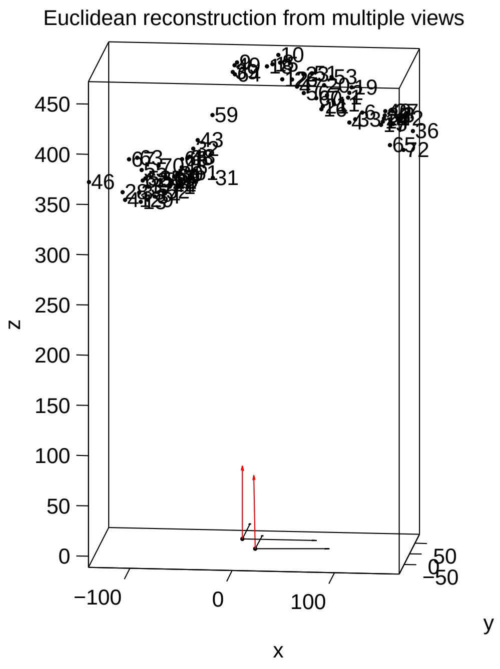


Figure 71: The estimated sparse depth map and camera pose for the selected feature points of the 1st and 88th frames.

Image registration is a fundamental task in computer vision and it significantly contributes to high-level computer vision and benefits numerous practical applications. Although there are already a lot of image registration techniques existing in literature, there is still a significant amount of research to be conducted because there are a lot of issues that need to be solved such as the parallax problem. The traditional image registration algorithms suffer from the parallax problem due to their underling assumption that the scene can be regarded approximately planar which is not satisfied when large depth variations exist in the images with high-rise objects. To address the parallax problem, we present a new strategy for 2D image registration by leveraging the depth information from 3D image reconstruction. The novel idea is to recover the depth in the image region with high-rise objects to build accurate transform function for image registration. We segment the 3D space in several separate regions and use surface fitting algorithms to estimate the 3D dense depth map. In order to segment the space geometrically, we propose a non-linear deterministic annealing algorithm for space partitioning. From the experimental results, the new method is able to mitigate the parallax problem and achieve robust image registration results. Our algorithm is attractive to numerous practical applications.

The results of Task 10 were reported in Ref. [132]. Next, we present the technical details.

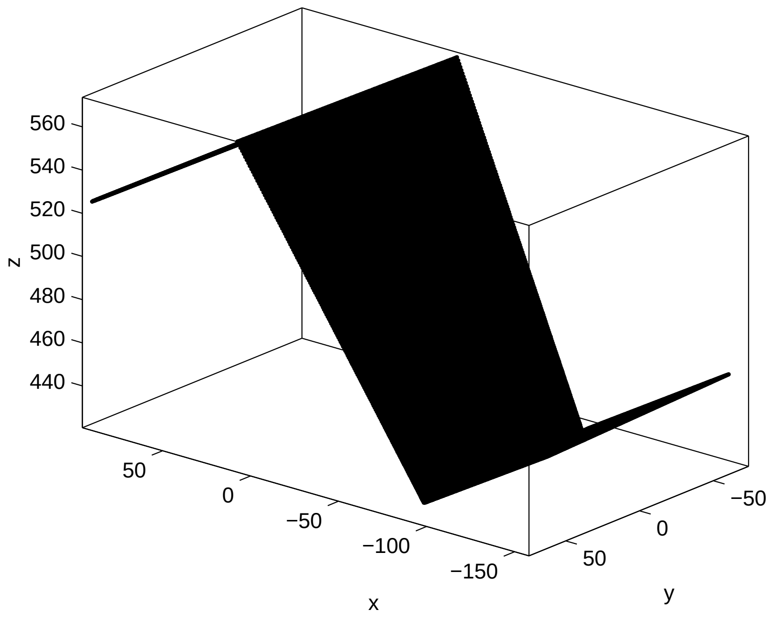


Figure 72: The estimated dense 3D configuration.

12.1 Introduction

Image registration is a fundamental task in image processing and computer vision which matches two or more images taken at different times and different viewpoints, by geometrically aligning reference and sensed images. There has been a broad range of techniques developed over the years in literature. A comprehensive survey of image registration methods was published in 1992 by Brown [62], including many classic methods still in use. Due to the rapid development of image acquisition devices, more image registration techniques emerged afterwards and were covered in another survey published in 2003 [63].

Different applications due to distinct image acquisition require different image registration techniques. In general, manners of the image acquisition can be divided into three main groups:

- *Different viewpoints (multiview analysis).* Images of the same scene are acquired from different viewpoints. The aim is to gain a larger 2D view or a 3D representation of the scanned scene.
- *Different times.* Images of the same scene are acquired at different times, often on regular basis, and possibly under different conditions. The aim is to find and evaluate changes in the scene which appeared between the consecutive image acquisitions.
- *Different sensors.* Images of the same scene are acquired by different sensors. The aim is to

integrate the information obtained from different source streams to gain more complex and detailed scene representation.

The prevailing image registration methods, such as Davis and Keck's algorithm [71, 72], assume all the feature points are coplanar and build a homography transform matrix to do registration. The advantage is that they have low computational cost and can handle planar scenes conveniently; however, with the assumption that the scenes are approximately planar, they are inappropriate in the registration applications when the images have large depth variation due to the high-rise objects, known as the parallax problem. Parallax is an apparent displacement of difference of orientation of an object viewed along two different lines of sight, and is measured by the angle or semi-angle of inclination between those two lines. Nearby objects have a larger parallax than further objects when observed from different positions. Therefore, as the viewpoint moves side to side, the objects in the distance appear to move slower than the objects close to camera.

In this section, we propose a depth based image registration algorithm by leveraging the depth information. Our method can mitigate the parallax problem caused by high-rise scenes in the images by building accurate transform function between corresponding feature points in multiple images. Given an image sequence, we first select a number of feature points and then match the features in all images. Then we estimate the depth of each feature point from feature correspondences. With the depth information, we can project the image in 3D instead of using a homography transform. Further more, fast and robust image registration algorithm can be achieved by combining the traditional image registration algorithms and depth based image registration method proposed in this section. The idea is that we first compute the 3D structure of a sparse feature points set and then divide the scene geometrically into several approximately planar regions. For each region, we can perform a depth based image registration. Accordingly, robust image registration is achieved.

The remainder of this section is organized as follows. We present the system scheme for 2D image registration in Section 12.2. Section 12.3 reviews the 3D reconstruction algorithm we used in our new method. In Section 12.4, we describe how to use 3D depth information for 2D image registration and propose a non-linear deterministic annealing algorithm for space partitioning. Section 12.5 presents the experimental results and we compare our algorithm with Davis and Keck's algorithm on the same test video sequence.

12.2 The scheme of the new 2D image registration system

Due to the diversity of images to be registered and various types of degradations, it is impossible to design a universal method applicable to all registration tasks. Every method should take

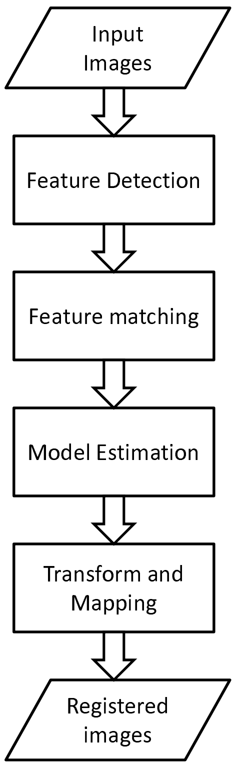


Figure 73: The pipeline for 2D image registration system.

into account not only the assumed type of geometric deformation between the images but also the radiometric deformations and noise corruption, required registration accuracy and application-dependent data characteristics. Nevertheless, the majority of the registration methods consists of the following four steps: feature detection, feature matching, transform model estimation, image resampling and transformation. Although they may differ in some specific part, most 3D reconstruction approaches are generally based on the same pipeline. The pipeline is given in Fig. 73.

A widely used feature detection method is corner detection. Kitchen and Rosenfeld [64] proposed to exploit the second-order partial derivatives of the image function for corner detection. Dreschler and Nagel [65] searched for the local extrema of the Gaussian curvature. However, corner detectors based on the second-order derivatives of the image function are sensitive to noise. Thus Forstner [66] developed a more robust, although time consuming, corner detector, which is based on the first-order derivatives only. The reputable Harris detector [67] also uses first-order derivatives for corner detection. Feature matching includes area-based matching and feature-based matching. Classical area-based method is cross-correlation (CC) [68] exploit for matching image intensities directly. For feature-based matching, Goshtasby [69] described the registration based on the graph matching algorithm. Clustering technique, presented by Stockman et al. [70], tries to match points connected by abstract edges or line segments. After the feature correspondence

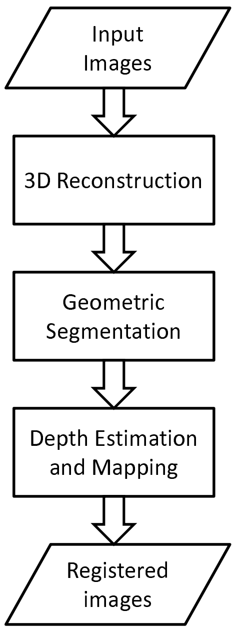


Figure 74: The new image registration system scheme.

has been established the mapping function is constructed. The mapping function should transform the sensed image to overlay it over the reference image. Finally interpolation methods such as nearest neighbor function, bilinear, and bicubic functions are applied to the output of the registered images.

In our new image registration system, we use a 3D model instead of 2D motion model used in existing works. The system scheme is slightly different from the previous one. We give the new scheme in Fig. 74. In the new system scheme, we first apply 3D reconstruction to the input images and recover the 3D geometric structure of the scene in the images. The 3D model is more accurate compared to the 2D motion models estimated in the previous works. Then we segment the 3D Euclidean space geometrically into several separate regions. Each region could be modeled by a linear plane. With the segmentation, we can estimate the 3D depth for every pixel in each region and recover the dense structure of the scene. The 3D dense structure enables the pixel by pixel mapping of the input images. We describe the 3D reconstruction algorithm in Section 12.3. In Section 12.4, we present the geometric segmentation and depth based mapping in 3D, and also propose a non-linear deterministic annealing algorithm for space partitioning.

12.3 3D reconstruction from video sequences

Here, we simply review the 3D reconstruction algorithm described in Ma et. al's book [73]. When developing a stereo vision algorithm for registration, the requirements for accuracy vary from those of standard stereo algorithms used for 3D reconstruction. For example, a multi-pixel disparity error in an area of low texture, such as a white wall, will result in significantly less intensity error in the registered image than the same disparity error in a highly textured area. In particular, edges and straight lines in the scene need to be rendered correctly.

The 3D reconstruction algorithm is implemented using the following steps. First, geometric features are detected automatically in each individual images. Secondly, feature correspondence is established across all the images. Then the camera motion is retrieved and the camera is calibrated. Finally the Euclidean structure of the scene is recovered.

12.3.1 Feature selection

The first step in 3D reconstruction is to select candidate features in all images for tracking across different views. Ma et al. [73] use point feature in reconstruction which is measured by Harris' criterion,

$$C(\mathbf{x}) = \det(G) + k \times \text{trace}^2(G) \tag{167}$$

where $\mathbf{x} = [x, y]^T$ is a candidate feature, $C(\mathbf{x})$ is the quality of the feature, k is a pre-chosen constant parameter and G is a 2×2 matrix that depends on $\mathbf{x}$, given by

$$G = \begin{bmatrix} \sum_{W(\mathbf{x})} I_x^2 & \sum_{W(\mathbf{x})} I_x I_y \\ \sum_{W(\mathbf{x})} I_x I_y & \sum_{W(\mathbf{x})} I_y^2 \end{bmatrix} \tag{168}$$

where $W(\mathbf{x})$ is a rectangular window centered at $\mathbf{x}$ and I_x and I_y are the gradients along the x and y directions which can be obtained by convolving the image I with the derivatives of a pair of Gaussian filters. The size of the window can be decided by the designer, for example 7×7. If $C(\mathbf{x})$ exceeds a certain threshold, then the point $\mathbf{x}$ is selected as a candidate point feature.

12.3.2 Feature correspondence

Once the candidate point features are selected, the next step is to match them across all the images. In this subsection, we use a simple feature tracking algorithm based on a translational model.

We use the sum of squared differences (SSD) [74] as the measurement of the similarity of two point features. Then the correspondence problem becomes looking for the displacement $\mathbf{d}$ that satisfies the following optimization problem:

$$\min_{\mathbf{d}} \doteq \sum_{\mathbf{x} \in W(\mathbf{x})} [I_2(\mathbf{x}+\mathbf{d}) - I_1(\mathbf{x})]^2 \tag{169}$$

where $\mathbf{d}$ is the displacement of a point feature of coordinates $\mathbf{x}$ between two consecutive frames I_1 and I_2. Lucas and Kanade also give the close form solution of 169

$$\mathbf{d} = -G^{-1}\mathbf{b} \tag{170}$$

where

$$\mathbf{b} \doteq \begin{bmatrix} \sum_{W(\mathbf{x})} I_x I_t \\ \sum_{W(\mathbf{x})} I_y I_t \end{bmatrix} \tag{171}$$

G is the same matrix we used to compute the quality of the candidate point feature in Eq. 167, and $I_t \doteq I_2 - I_1$.

12.3.3 Estimation of camera motion parameters

In this subsection, we recover the projective structure of the scene from the established feature correspondence. We will follow the notation used in Ma et al.'s book [73]. For the detail of the proof of this algorithm, please refer to the reference.

The reconstruction algorithm is based on a perspective projection model with a pinhole camera. Suppose we have a generic point $p \in \mathbb{E}^3$ with coordinates $\mathbf{X} = [X, Y, Z, 1]^T$ relative to a world coordinate frame. Given two frames of one scene which is related by a motion $g = (R, T)$, the two image projection point $\mathbf{x}_1$ and $\mathbf{x}_2$ are related as follows:

$$\lambda_1 \mathbf{x}'_1 = \Pi_1 \mathbf{X}_p, \quad \lambda_2 \mathbf{x}'_2 = \Pi_2 \mathbf{X}_p \tag{172}$$

where $\mathbf{x}' = [x, y, 1]^T$ is measured in pixels, λ_1 and λ_2 are the depth scale of $\mathbf{x}_1$ and $\mathbf{x}_2$, $\Pi_1 = [K, 0]$ and $\Pi_2 = [KR, KT]$ are the camera projection matrices and K is the camera calibration matrix. In order to estimate λ_1, λ_2, Π_1 and Π_2, we need to introduce the epipolar constraint. From Eq. 172, we have

$$\mathbf{x}'^T_2 K^{-T} \hat{T} R K^{-1} \mathbf{x}'_1 = 0 \tag{173}$$

The fundamental matrix is defined as:

$$F \doteq K^{-T} \hat{T} R K^{-1} \tag{174}$$

With the above model, we could estimate the fundamental matrix F via the Eight-point algorithm [73]. Then we could decompose the fundamental matrix to recover the projection matrices Π_1 and Π_2 and the 3D structure. We only give the solution here by canonical decomposition:

$$\Pi_1 p = [I, 0], \Pi_2 p = [(\widehat{T'})^T F, T'], \lambda_1 \mathbf{x}'_1 = \mathbf{X}_p, \lambda_2 \mathbf{x}'_2 = (\widehat{T'})^T F \mathbf{X}_p + T' \tag{175}$$

Table 14: Eight-point algorithm

Given a set of initial point feature correspondences expressed in pixel coordinates $(\mathbf{x'}_1^j, \mathbf{x'}_2^j)$ for $j = 1, 2, ..., n$:

• **A first approximation of the fundamental matrix:** Construct the matrix $\chi \in \mathbb{R}^{n\times 9}$ from the transformed correspondences $\tilde{\mathbf{x}}_1^j \doteq [\tilde{x}_1^j, \tilde{y}_1^j, 1]^T$ and $\tilde{\mathbf{x}}_2^j \doteq [\tilde{x}_2^j, \tilde{y}_2^j, 1]^T$, where the jth row of χ is given by $[\tilde{x}_1^j\tilde{x}_2^j, \tilde{x}_1^j\tilde{y}_2^j, \tilde{x}_1^j, \tilde{y}_1^j\tilde{x}_2^j, \tilde{y}_1^j\tilde{y}_2^j, \tilde{y}_1^j, \tilde{x}_2^j, \tilde{y}_2^j, 1]^T \quad \in \mathbb{R}^9$. Find the vector $F^s \in \mathbb{R}^9$ of unit length such that $||\chi F^s||$ is minimized as follows: Compute the singular value decomposition (SVD) of $\chi = U\Sigma V^T$ and define F^s to be the ninth column of V. Unstack the nine elements of F^s into a square 3×3 matrix $\tilde{F}$.

• **Imposing the rank-2 constraint:** Compute the SVD of the matrix F recovered from data to be $\tilde{F} = U_F diag\{\sigma_1, \sigma_2, \sigma_3\}V_F^T$. Impose the rank-2 constraint by letting $\sigma_3 = 0$ and reset the fundamental matrix to be $F = U_F diag\{\sigma_1, \sigma_2, 0\}V_F^T$.

12.3.4 Depth estimation

The Euclidean structure $\mathbf{X}_e$ is related to the projective reconstruction $\mathbf{X}_p$ by a linear transform $H \in \mathbb{R}^{4\times 4}$,

$$\Pi_{ip} \sim \Pi_{ie}H^{-1}, \mathbf{X}_p \sim H\mathbf{X}_e, i = 1, 2, ..., m \tag{176}$$

where $\sim$ means equality up to a scale factor and

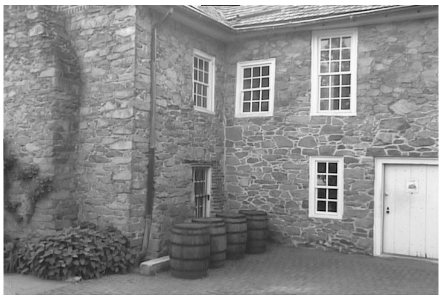
$$H = \begin{bmatrix} K_1 & 0 \\ -\nu^T K_1 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{177}$$

With the assumption that K is constant, we could estimate the unknowns K and ν with a gradient decent optimization algorithm. In order to obtain a unique solution, we also assume that the scene is generic and the camera motion is rich enough.

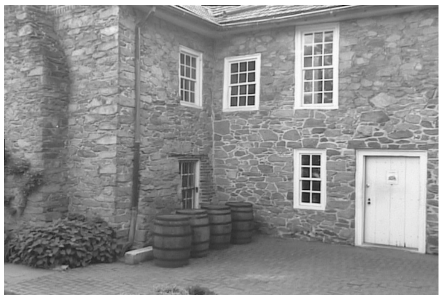
Fig. 75 shows the first frame and the 88th frame of the test video sequence 'oldhousing'. In our experiment, we will register all the frames in the video sequence to the first frame. Fig. 76 show the selected feature points on the first frame which are used for camera pose estimation. Fig. 77 show the estimated depth map of the selected feature points and the camera pose.

12.4 Image registration with depth information

Once we obtain the 3D structure of the feature points, the motion, and calibration of the camera, we can start to register the rest of the pixels in the images with the estimated depth information. The



(a) The 1st frame in the ‘oldhousing’ video sequence



(b) The 88th frame in the ‘oldhousing’ video sequence

Figure 75: Original frames used for image registration

traditional image registration algorithms, such as the algorithm proposed by Davis and Keck [71, 72], try to register the two images by computing the homography matrix H between corresponding feature points. The limit of this algorithm is that they assume all the points in the physical world are coplanar or approximately coplanar, which is not true with high-rise scenes. In order to mitigate this problem, we propose a novel algorithm which first segment the image geometrically and then perform the registration to each region with depth estimation.

12.4.1 Geometrical segmentation

In order to perform the geometrical segmentation, the most intuitive method is to obtain the dense surface model of the scene and then segment the surface into several regions based on the depth of the points. However, we need to know the correspondence for almost all the pixels to compute the dense surface model, which means we need to know all the pixel correspondence before the registration. In order to avoid this dilemma, we will not use the traditional 3D reconstruction algorithm to estimate the dense surface model. Instead, we directly segment the scene into several regions by clustering the sparse 3D points set that we obtained in Section 12.3. With the assumption that each segment region of the scene is approximately coplanar in the physical world, we could easily estimate the plane model and project the 3D plane onto the image frames. Comparing the assumption that the whole scene is coplanar in the physical world used in the traditional image registration algorithms, this assumption is valid in most circumstances.

There are a lot of algorithms for data clustering. The most famous hard-clustering algorithm is k-means [75]. The k-means algorithm assigns each data point to the cluster whose centroid is nearest. Here, we use the distance to a 3D plane in the physical world as the measurement.

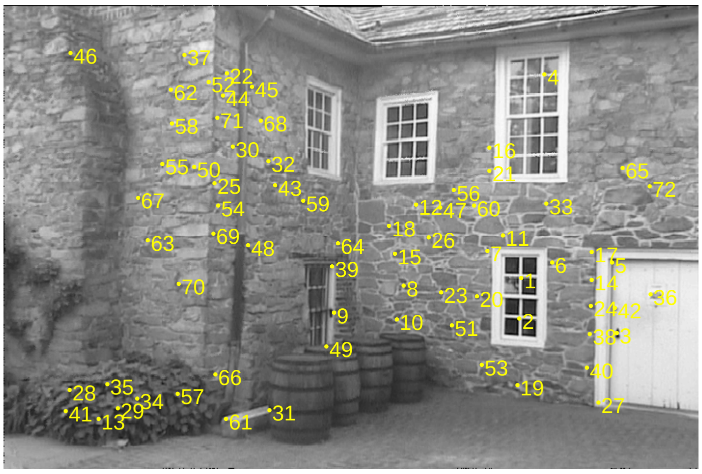


Figure 76: The feature points selected for depth estimation on the 1st frame.

For each cluster, we could choose the plane that has the smallest sum of distance of all the data points in the cluster. However, the descent based learning methods suffer from a serious limitation. The non-global optima of the cost surface may easily resulting in poor local minima to the above methods. Techniques adding penalty terms to the cost function further increases the complexity of the cost surface and worsen the local minimum problem.

In this section, we propose a non-linear deterministic annealing approach to solve the 3D geometrical fitting problem. We follow the deterministic annealing approach [104] and use the geometrical structure for clustering. Deterministic Annealing introduce the entropy constraint to explore a large portion of the cost surface using randomness, while still performing optimization using local information, which is similar to fuzzy c-means algorithm. Eventually, the amount of imposed randomness is lowered so that upon termination DA optimizes over the original cost function and yields a solution to the original problem.

To solve the space partitioning problem, we do not use prototype to calculate the difference. The reason is that the prototype in space partitioning is generally not sufficient to represent a plane in 3D space. Instead, we estimate the linear plane model and calculate the fitting error as the Euclidean distance between the data and the plane. The traditional local optimization algorithm will likely stuck at a local optima. In order to avoid local optima, we use local geometric structure from neighboring data points and embedded the data vectors to a higher dimension as follows.

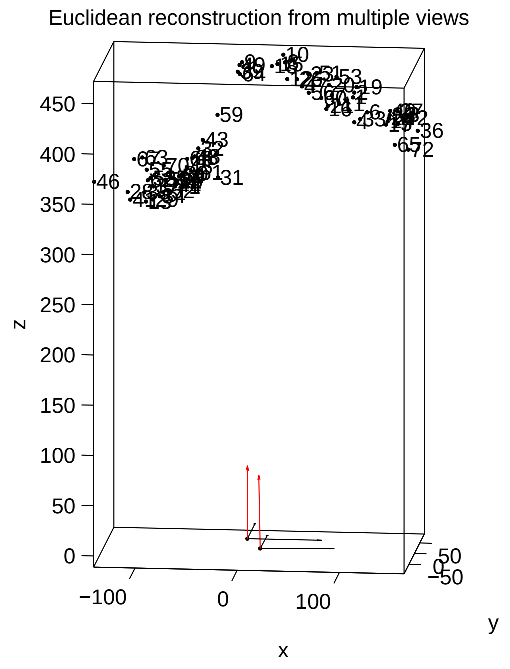


Figure 77: The estimated depth map and camera pose for the selected feature points of the 1st and 88th frames.

The input data is given as a 3D point, $\mathbf{x}_i = [x_i, y_i, z_i]^T$. With the assumption that nearest data points are on the same plane, we could estimate the local plane model, $\mathbf{L}_i = [a_i, b_i, c_i]^T$ of data point $\mathbf{x}_i$ and its K nearest neighbor points.

$$\mathbf{L} = \begin{bmatrix} a(\mathfrak{X}) \\ b(\mathfrak{X}) \\ c(\mathfrak{X}) \end{bmatrix} \tag{178}$$

$$\mathbf{f} = \begin{bmatrix} \mathbf{x} \\ \mathbf{L} \end{bmatrix} \tag{179}$$

Then we revise the distortion function as follows,

$$D(\mathbf{f}_i, g_{\theta_j}) = D_1(I_1\mathbf{f}_i, g_{\theta_j}) + D_2(I_2\mathbf{f}_i, g_{\theta_j}) \tag{180}$$

$$I_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{181}$$

$$I_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{182}$$

where $D_1 = d_{i,j}$ calculate the fitting error between the data point and the estimated plane, and D_2 calculate the difference between the local estimated plane model and the cluster scale estimated plane model. D_2 is defined as follows:

$$D_2(I_2\mathbf{f}_i, g_{\theta_j}) = \frac{I_2\mathbf{f}_i^T \times g_{\theta_j}}{|I_2\mathbf{f}_i| \times |g_{\theta_j}|} \tag{183}$$

After the mapping, we apply deterministic annealing algorithm to partition the data into several clusters as follows.

$$\min_{g_{\theta_j}} F = D - TH \tag{184}$$

where $g_{\theta_j} = [a_j, b_j, c_j]$ is the geometrical surface model parameter to be estimated, D is the sum of square of geometrical fitting error and H is the entropy constraint. We define D and H as follows:

$$D = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j}) d(\mathbf{x}_i, g_{\theta_j}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i) d(\mathbf{x}_i, g_{\theta_j}) \tag{185}$$

$$H(\mathbf{X}, g_\theta) = \sum_{i=1}^{N}\sum_{j=1}^{K} p(\mathbf{x}_i, g_{\theta_j}) \log p(\mathbf{x}_i, g_{\theta_j}) \tag{186}$$

To perform optimization we need to further analyze its terms. We can rewrite equation (186) by applying the chain rule of entropy as

$$H(\mathbf{X}, g_\theta) = H(\mathbf{X}) + H(g_\theta|\mathbf{X}) \tag{187}$$

Notice that the first term $H(\mathbf{X})$ is the entropy of the source and is therefore constant with respect to the cluster g_{θ_j} and association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$. Thus we can just focus on the conditional entropy

$$H(g_\theta|\mathbf{X}) = \sum_{i=1}^{N} p(\mathbf{x}_i) \sum_{j=1}^{K} p(g_{\theta_j}|\mathbf{x}_i) \log p(g_{\theta_j}|\mathbf{x}_i) \tag{188}$$

The minimization of F with respect to association probabilities $p(g_{\theta_j}|\mathbf{x}_i)$ gives rise to the Gibbs distribution

$$p(g_{\theta_j}|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T})}{Z_x} \tag{189}$$

where the normalization is

$$Z_x = \sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i, g_{\theta_j})}{T}) \tag{190}$$

The corresponding minimum of F is obtained by plugging equation (189) back into equation (184)

$$F^* = \min_{p(g_{\theta_j}|\mathbf{x}_i)} F = -T \sum_{i=1}^{N} p(\mathbf{x}_i) \log Z_x \tag{191}$$

To minimize the Lagrangian with respect to the cluster model g_{θ_j}, its gradients are set to zero yielding the condition

$$\nabla_{g_{\theta_j}} F = \frac{1}{N} \sum_{i=1}^{N} p(g_{\theta_j}|\mathbf{x}_i) \nabla_{g_{\theta_j}} d(\mathbf{x}_i, g_{\theta_j}) = 0 \tag{192}$$

Non-linear deterministic annealing method (NDA) introduces the entropy constraint to explore a large portion of the cost surface using randomness, while still performing optimization using local information, which is similar to fuzzy c-means algorithm. Eventually, the amount of imposed randomness is lowered so that upon termination NDA optimizes over the original cost function and yields a solution to the original problem.

However, there is no close form solution, therefore we use a gradient descent algorithm to solve this problem. I present our algorithm in Figure. 78.

12.4.2 Depth estimation

Here, we only consider two images. Suppose for the first image, we have the 3D point set $\mathbf{X}_e^j, j = 1, 2, ..., n$ which could be divided into three clusters,$\mathbf{X}_{e1}$, $\mathbf{X}_{e2}$, $\mathbf{X}_{e3}$. For each cluster, there are at least three non-collinear points. Then we could have the plane model for this cluster. Let's take the example of $\mathbf{X}_{e1}$, suppose there are m points in the cluster and we have the plane model as follows:

$$\mathbf{A} \cdot p = 1. \tag{195}$$

where $\mathbf{A} = [\mathbf{X}_{e1}^i], i = 1, ..., m$ and $p = [a, b, c]^T$ is the plane parameter.

Given an arbitrary point $\mathbf{x}^i = [x^i, y^i]^T$ measured in pixels in the first cluster, we could estimate it's depth scale λ^i by solving the following equation.

$$\lambda^i \mathbf{x}'^i = H_1^{-1} \Pi_1 \mathbf{X}_e^i. \tag{196}$$

```
1.  Algorithm 78 NDA based geometrical segmentation algorithm
2.  Set Limit
3.      $K_{max}$: maximum number of clusters
4.      $T_{init}$: starting temperature
5.      $T_{min}$: minimum temperature
6.      $\delta$: perturbation vector
7.      $\alpha$: cooling rate (must be $< 1$)
8.      $I_{max}$: maximum iteration number
9.      $th$: Iteration threshold
10.     $sth$: Surface distance threshold
11. Initialization
12.     $T = T_{init}, K = 2, \Lambda_1 = (X^T X)^{-1} X^T \vec{1}, \Lambda_2 = \Lambda_1, [p(\mathbf{\Lambda}_1|\mathbf{x}_i), p(\mathbf{\Lambda}_2|\mathbf{x}_i)] = [\frac{1}{2}, \frac{1}{2}], \forall i.$
13. Perturb
14.     $\Lambda_j = \Lambda_j + \delta, \forall j.$
15.     $L_{old} = D - TH.$
16. Loop until convergence, $i = 0 \; \forall j$
17.     For all $\mathbf{x}_i$ in the training data, compute the association probabilities
            $p(\mathbf{\Lambda}_j|\mathbf{x}_i) = \frac{\exp(-\frac{d(\mathbf{x}_i, \mathbf{\Lambda}_j)}{T})}{\sum_{j=1}^{K} \exp(-\frac{d(\mathbf{x}_i, \mathbf{\Lambda}_j)}{T})}$    (193)
18.     update the surface model
            $\mathbf{\Lambda}_j \longleftarrow \mathbf{\Lambda}_j + \alpha \nabla_{\mathbf{\Lambda}_j} F.$    (194)
19.     i = i+1;
20.     if ($i > I_{max}$ or $\nabla_{\mathbf{\Lambda}_j} F < th$ ) End Loop
21. Model Size Determination
22.     if($d(\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}) < sth$)
23.         replace $\mathbf{\Lambda}_j, \mathbf{\Lambda}_{j+1}$ by a single plane
24.     $K$ =number of planes after merging
25. Cooling Step
26.     $T = \alpha T.$
27.     if ($T < T_{min}$)
28.         perform last iteration for $T = 0$ and STOP
29. Duplication
30.     Replace each plane by two planes at the same location, $K = 2K$.
31. Goto Step 10
```

where $\mathbf{x}'^i = [x^i, y^i, 1]^T$, H_1^{-1} and Π_1 are estimated in Section 12.3. In Eq. 196, only λ^i is unknown and with the constraint on $\mathbf{X}_e^i$ with Eq. 195, we could easily get the value of λ^i.

Then, with $\Pi_1 = [I, 0]$, we could have $X_p^i = [\lambda_1^i x^i, \lambda_1^i y^i, \lambda_1^i, 1]$. from Eq. 172, we can get the relation between two image projection point $\mathbf{x}_1^i$ and $\mathbf{x}_2^i$ as follows:

$$\widehat{\mathbf{x_2^i}'} = \Pi_2 \mathbf{X}_p^i. \tag{197}$$

where $\widehat{\mathbf{x_2^i}'} = [\lambda_2^i x_2^i, \lambda_2^i y_2^i, \lambda_2^i]$. We could then get the position of the corresponding point $\mathbf{x}_2^i = [x_2^i, y_2^i]$ in the second image.

12.5 Experimental Results

The data includes a sequence of 88 images captured from one camera. We first select 72 feature points in the first image and then find the corresponding feature points in the rest of the images. The depth estimates of these points are calculated by the algorithm introduced in Section 12.3.

In our experiment, we regard the first image's local coordinate system as world coordinate system so the first image can be viewed as a reference image. Then the rest of the images are registered to the reference image. We also applied the algorithm proposed by Davis and Keck [71] to register the input images for comparison purpose.

Fig. 75 is the 1st frame and the 88th frame in the test image sequence. Fig. 79 is the registration result using our algorithm and Fig. 80 is the output of the algorithm proposed by Davis and Keck [71]. Fig. 81 shows the difference image between the registered image and the first image using our algorithm and Fig. 82 shows the difference image from the algorithm of Davis and Keck [71]. We can see that our result can mitigate the parallax problem since the roof and wall corners are registered correctly; on the contrary, the registered image by the algorithm of Davis and Keck [71] has a lot of artifacts caused by the parallax problem. We also show some registration results using our algorithm in Fig. 83$\sim$ Fig. 84.

In order to further compare our algorithm to the algorithm proposed by Davis and Keck, we compute the root of mean squared errors (RMSE) of the registration results from both algorithms. Fig. 85 shows that the registration error of our algorithm is less than 50% than that of the algorithm proposed by Davis and Keck.

The result shows that our image registration algorithm can mitigate the parallax problem because most of the scene is registered without vibration, as opposed to registration results under the algorithm of Davis and Keck in which the high-rise scene in the sensed images significantly

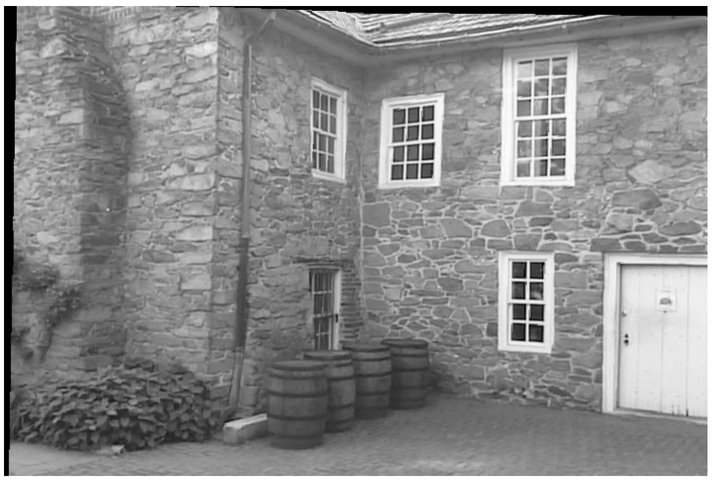


Figure 79: Our algorithm test result, in which the 88th frame is registered to the 1st frame.

moved after registration to the reference images. The reason is that the algorithm of Davis and Keck assumes all the points in the images are coplanar. While this assumption is satisfied when the distance between the camera and the interested scene is so large that the small depth variation can be neglected, it fails in the case of high-rise scene. Therefore, depth information should be used to accomplish the registration for this specific high-rise region of the images.

Finally, we would like to point out that the algorithm proposed by Davis and Keck [71] assumes a planar registration. Their scheme was designed for use with high-altitude aerial imagery where planar transformations are fairly good approximations. Furthermore, their scheme uses RANSAC to remove poor matching points during the computation. This can help to deal with some depth discontinuities that may be present in the high-altitude aerial images. In our experiments, the test images contain salient 3D scenes; these images are out of the domain for the algorithm of Davis and Keck. This is the reason why the algorithm of Davis and Keck does not perform well.

13 Conclusions

In this project, we have accomplished the research objective of developing advanced techniques for scene analysis, and have completed the ten research tasks. Specifically, we have developed the following new techniques:

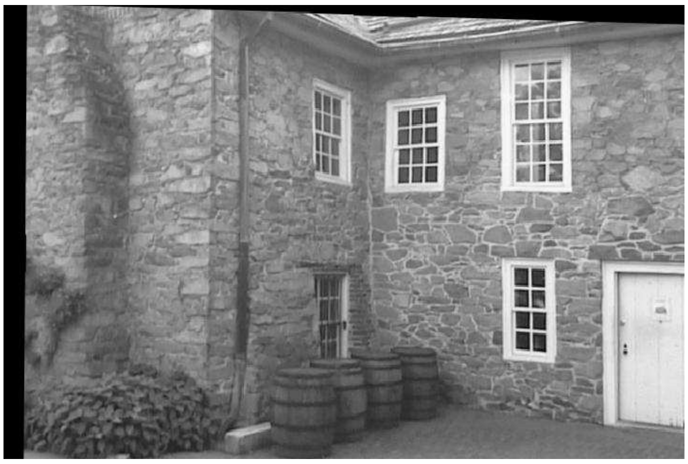


Figure 80: The test result under the algorithm of Davis and Keck, in which the 88th frame is registered to the 1st frame.

- robust feature-based algorithm for object tracking,
- motion-segmentation-based technique for change detection,
- a target detection algorithm that consists of image differencing, maximum-margin classifier, and diversity combining,
- a rotation-invariant transform for change detection,
- a depth-based image registration algorithm,
- an image registration algorithm that leverages wavelet,
- a machine learning algorithm to automatically recover 3D surface from sparse 3D points,
- an automatic surface fitting method for 3D reconstruction from 2D video sequence,
- a depth-based image registration method via geometric segmentation.

References

[1] B. Han, W. Roberts, D. Wu, and J. Li, "Robust feature-based object tracking," in *Proceedings of SPIE Defense & Security Symposium*, vol. 6568, Orlando, FL, USA, April 2007.

[2] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of Computer Vision and Pattern Recognition (CVPR 1994)*, Seattle, USA, June 1994, pp. 593–600.

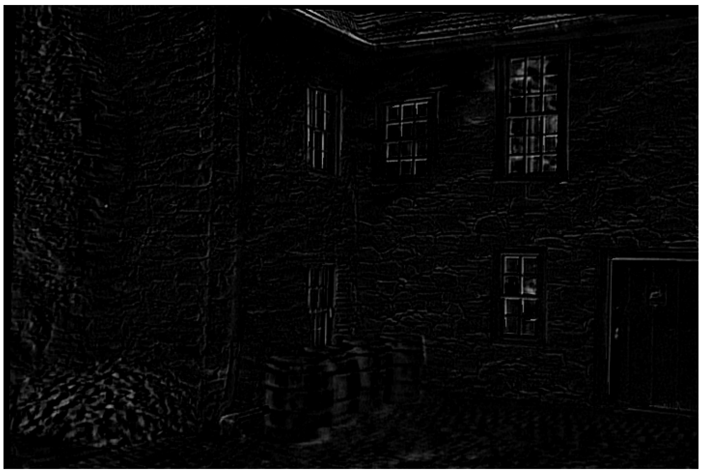


Figure 81: The difference image between the registered 88th image (using our algorithm) and the 1st image.

[3] P. F. H. Jin and S. Soatto, "Real-time feature tracking and outlier rejection with changes in illumination," in *Proceedings of International Conference on Computer Vision (ICCV 2001)*, July 2001, pp. 684–689.

[4] M. W. H. T. Nguyen and R. V. D. Boomgaard, "Occlusion robust adaptive template tracking," in *Proceedings of International Conference on Computer Vision (ICCV 2001)*, July 2001, pp. 678–683.

[5] J. G. Legters and T. Young, "A mathematical model for computer image tracking," in *Proceedings of Pattern Analysis and Machine Intelligence (PAMI 1982)*, Nov 1982, pp. 583–594.

[6] C. Tomasi and T. Kanade, "Factoring image sequences into shape and motion," in *Proceedings of the IEEE Workshop on Visual Motion*, Princeton, USA, Oct 1991, pp. 21–28.

[7] C. K. I. Williams and M. K. Titsias, "Fast unsupervised greedy learning of multiple objects and parts from video," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2004, pp. 79–87.

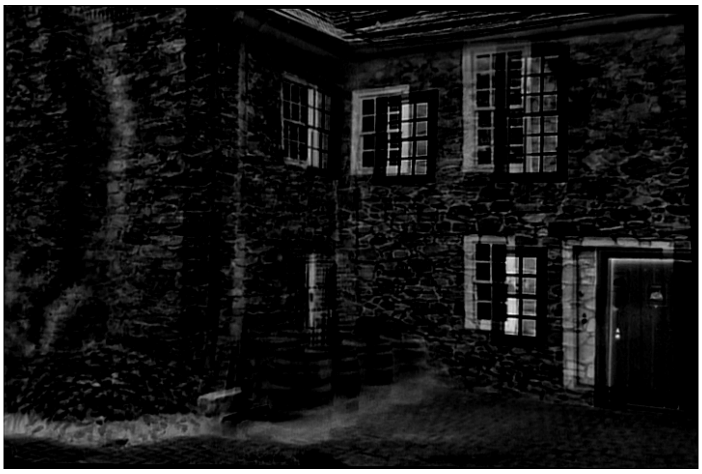


Figure 82: The difference image between the registered 88th image (using the algorithm of Davis and Keck) and the 1st image.

[8] N. Jojic and B. Frey, "Learning flexible sprites in video layers," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, 2001, pp. I:199–206.

[9] R. M. Neal and G. E. Hinton, "A view of the em algorithm that justifies incremental, sparse and other variants," in *Learning in Graphical Models.* Kluwer Academic, 1998, pp. 355–368.

[10] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of Image Understanding Workshop*, April 1981, pp. 121–130.

[11] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," in *Proceedings of International Conference on Image Processing (ICIP 1995)*, Oct 1995, pp. 444–447.

[12] B. Han, W. Roberts, D. Wu, and J. Li, "Motion-segmentation-based change detection," in *Proceedings of SPIE Defense & Security Symposium*, vol. 6568, Orlando, FL, USA, April 2007.

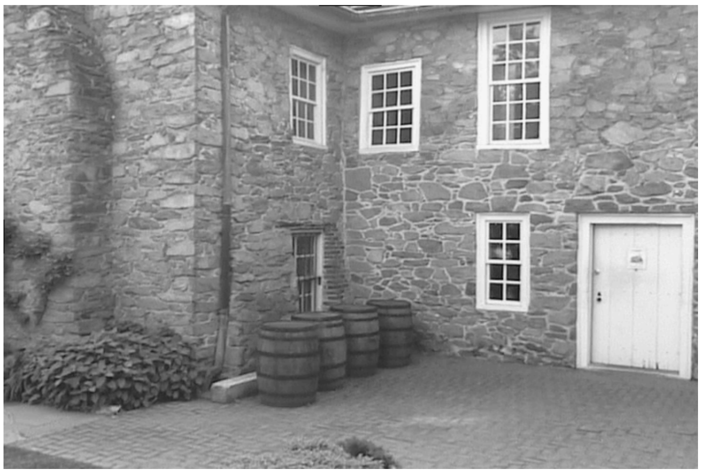


Figure 83: The 37th frame in the 'oldhousing' video sequence.

[13] P. Bouthemy and E. Francois, "Motion segmentation and qualitative dynamic scene analysis from an image sequence," in *International Journal of Computer Vision*, April 1993, pp. 157–182.

[14] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," in *Pattern Recognition*, 1993, pp. 1277–1294.

[15] D. W. Murray and B. F. Buxton, "Scene segmentation from visual motion using global optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 2, pp. 220–228, 1987.

[16] D. L. Gall, "Mpeg: a video compression standard for multimedia applications," *Commun. ACM*, vol. 34, no. 4, pp. 46–58, 1991.

[17] W. Aref, M. Hammad, A. C. Catlin, I. Ilyas, T. Ghanem, A. Elmagarmid, and M. Marzouk, "Video query processing in the vdbms testbed for video database research," in *MMDB '03: Proceedings of the 1st ACM international workshop on Multimedia databases*. New York, NY, USA: ACM Press, 2003, pp. 25–32.

[18] R. Pless and D. Jurgens, "Road extraction from motion cues in aerial video," in *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*. New York, NY, USA: ACM Press, 2004, pp. 31–38.

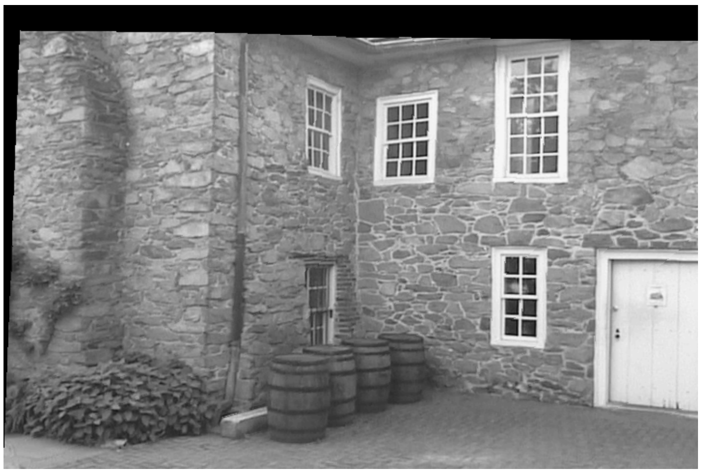


Figure 84: Our algorithm test result, in which the 37th frame is registered to the 1st frame.

[19] B. K. P. Horn and B. G. Schunck, “Determining optical flow,” in *Artificial Intelligence,*, July 1981, pp. 185–203.

[20] B. Lucas and T. Kanade, “Performance of optical flow techniques,” in *Proc. DARPA IU Workshop*, 1981, pp. 121–130.

[21] J. J. Y. A. Wang and E. H. Adelson, “Representing moving images with layers,” in *Image Processing, IEEE Transactions on*, Sep 1994, pp. 625–638.

[22] Y. A. G. D. Borshukov, G. Bozdagi and A. M. Tekalp, “Motion segmentation by multistage affine classification,” in *Image Processing, IEEE Transactions on*, Nov 1997, pp. 1591–1594.

[23] D. Cremers and C. Schnorr, “Motion competition: Variational integration of motion segmentation and shape regularization,” in *Proceedings of Pattern Recognition*, Zurich, Switzerland, Sep 2002, pp. 472–480.

[24] Y. Ma, H. Derkesen, W. Hong, and J. Wright, “Segmentation of multivariate mixed data via lossy coding and compression,” *to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[25] W. Hong, “Hybrid models for representation of imagery data,” Ph.D. dissertation, University of Illinois at Urbana-Champaign, Aug. 2006.

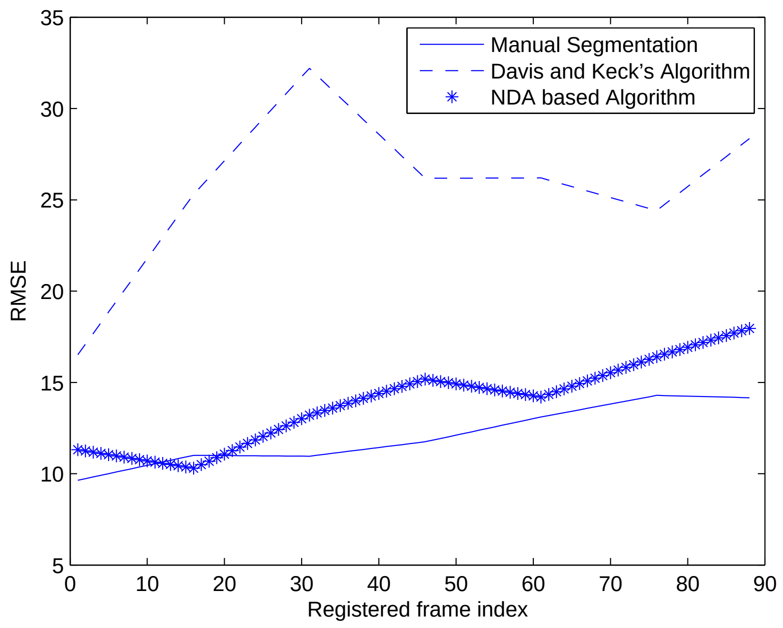


Figure 85: Our algorithm test result comparing to that under the algorithm of Davis and Keck, in which all the 88 frames are registered to the 1st frame.

[26] W. Roberts, L. Watkins, D. Wu, and J. Li, "Vehicle tracking for urban surveillance," in *Proceedings of SPIE Defense & Security Symposium*, vol. 6970, Orlando, FL, USA, March 2008.

[27] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," ser. Proc. of Image Understanding Workshop, 1981, pp. 121–130.

[28] C. Tomasi and T. Kanade, "Factoring image sequences into shape and motion," ser. Proc. of the IEEE Workshop on Visual Motion, 1991, pp. 21–28.

[29] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of the ASME - Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[30] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *Trans. of the ASME - Journal of Basic Engineering*, vol. 83, pp. 95–107, 1961.

[31] G. L. Jr. and T. Young, "A mathematical model for computer image tracking," *IEEE Trans. on PAMI*, vol. 4, no. 6, pp. 583–594, 1982.

[32] N. Papanikolopoulos, P. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision," *IEEE Trans. on Robotics and Automation*, vol. 9, pp. 14–35, 1993.

[33] M. Worring, H. T. Nguyen, and R. V. D. Boomgaard, "Occlusion robust adaptive template tracking," ser. Proc. of International Conf. on Computer Vision, 2001, pp. 678–683.

[34] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research: Part C*, vol. 6, no. 4, pp. 271–288, 1998.

[35] S. Gupte, O. Masoud, R. F. K. Martin, and N. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Trans. on Intell. Transport. Syst.*, vol. 3, no. 1, pp. 37–47, 2002.

[36] S. Chen, M. Shyu, S. Peeta, and C. Zhang, "Spatiotemporal vehicle tracking: the use of unsupervised learning-based segmentation and object tracking," *IEEE Robotics and Automation Magazine*, vol. 12, no. 1, pp. 50–58, 2005.

[37] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *IEEE Trans. on Intell. Transport. Syst.*, vol. 1, no. 2, pp. 108–118, 2000.

[38] W. Ye, C. Paulson, D. Wu, and J. Li, "A target detection scheme for VHF SAR ground surveillance," in *Proceedings of SPIE Defense & Security Symposium*, vol. 6970, Orlando, FL, USA, March 2008.

[39] B. Binder, M. Toups, S. Ayasli, and E. Adams, "SAR foliage penetration phenomenology of tropical rain forest and northern US forest," *Proceedings of IEEE International Radar Conference, 1995, Alexandria, VA*, pp. 158–163, 8-11 May 1995.

[40] L. Bessette and S. Ayasli, "Ultra-wideband P-3 and CARABAS II foliage attenuation and backscatter analysis," *Proceedings of 2001 IEEE Radar Conference, Atlanta, GA*, pp. 357–362, 1-3 May 2001.

[41] Y. Sun, Z. Liu, S. Todorovic, and J. Li, "Adaptive boosting for synthetic aperture radar automatic target recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 112–125, January 2007.

[42] L. Kaplan, "Improved SAR target detection via extended fractal features," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 2, pp. 436–451, April 2001.

[43] D. Howard, S. Roberts, and R. Brankin, "Target detection in SAR imagery by genetic programming," *Advances in Engineering Software*, vol. 30, no. 5, pp. 303–311, May 1999.

[44] T. Cooke, N. Redding, J. Schroeder, and J. Zhang, "Comparison of selected features for target detection in synthetic aperture radar imagery," *Conference record of the 33rd Asilomar Conference on Signals, Systems, and Computers*, vol. 2, pp. 859–863, 1999.

[45] S. Kuttikkad and R. Chellappa, "Non-Gaussian CFAR techniques for target detection in high resolution SAR images," *Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 910–914, 1994.

[46] M. Lundberg, L. Ulander, W. Pierson, and A. Gustavsson, "A challenge problem for detection of targets in foliage," *Proceedings of SPIE*, 2006.

[47] L. Ulander, P. Frolind, A. Gustavsson, H. Hellsten, and B. Larsson, "Detection of concealed ground targets in CARABAS SAR images using change detection," *Proceedings of SPIE - International Society Optical Engineering*, vol. 3721, pp. 243–252, 1999.

[48] L. Ulander, B. Flood, P. Follo, P. Frolind, A. Gustavsson, T. Jonsson, B. Larsson, M. Lundberg, W. Pierson, and G. Stenstrom, "Flight Campaign Vidsel 2002, CARABAS-II change detection analysis," no. FOI-R-1001-SE, 2003.

[49] L. Ulander, W. Pierson, M. Lundberg, and A. Gustavsson, "Performance of VHF-band SAR change detection for wide-area surveillance of concealed ground targets," *Proceedings of SPIE - International Society Optical Engineering*, vol. 5427, pp. 259–270, 2004.

[50] L. Ulander, W. Pierson, M. Lundberg, P. Follo, P. Frolind, and A. Gustavsson, "CARABAS-II SAR change detection performance on ground targets concealed in foliage," *Proceedings of EUSAR 2004, 5th European Conference on Synthetic Aperture Radar, Ulm, GE*, pp. 297–300, 25-27 May 2004.

[51] "https://www.sdms.afrl.af.mil/datasets/vhf_change_detection/index.php."

[52] L. Ulander, M. Lundberg, W. Pierson, and A. Gustavsson, "Change detection for low-frequency SAR ground surveillance," *IEE Proceedings of Radar Sonar Navigation*, vol. 152, no. 6, pp. 413–420, December 2005.

[53] W. Hong, "Hybrid models for representation of imagery data," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2006.

[54] K. Kira and L. Rendell, "A practical approach to feature selection," *Proceedings of the 9th International Workshop on Machine Learning*, pp. 249–256, 1992.

[55] Y. Sun and J. Li, “Iterative RELIEF for feature weighting,” *Proceedings of the 23rd ACM International Conference on Machine learning*, vol. 148, pp. 913–920, 2006.

[56] W. Ye, C. Paulson, D. Wu, and J. Li, “A rotation-invariant transform for target detection in SAR images,” in *Proceedings of SPIE Defense & Security Symposium*, vol. 6970, Orlando, FL, USA, March 2008.

[57] ——, “A target detection scheme for VHF SAR ground surveillance,” *Proceedings of SPIE Defense and Security Symposium 2008, Orlando, FL*, 2008.

[58] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[59] D. Lowe, “Object recognition from local scale-invariant keypoints,” *Proceedings of the 7th International Conference on Computer Vision*, pp. 1150–1157, 1999.

[60] R. Bracewell, *Two-Dimensional Imaging*. Englewood Cliffs, NJ: Prentice Hall, 1995.

[61] B. Han, C. Paulson, J. Wang, and D. Wu, “Depth-based image registration,” in *Proceedings of SPIE Defense & Security Symposium*, vol. 7699, Orlando, FL, USA, April 2010.

[62] L. Brown, “A survey of image registration techniques,” *ACM computing surveys (CSUR)*, vol. 24, no. 4, pp. 325–376, 1992.

[63] B. Zitova and J. Flusser, “Image registration methods: a survey,” *Image and vision computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[64] L. Kitchen and A. Rosenfeld, “Gray-level corner detection,” 1980.

[65] L. Dreschler and H. Nagel, *Volumetric model and 3D-trajectory of a moving car derived from monocular TV-frame sequences of a street scene*. Univ., Fachbereich Informatik, 1981.

[66] W. Forstner and E. Gulch, “A fast operator for detection and precise location of distinct points, corners and centres of circular features,” in *Proc. ISPRS Intercommission Conference on Fast Processing of Photogrammetric Data*, 1987, pp. 281–305.

[67] J. Noble, “Finding corners,” *Image and Vision Computing*, vol. 6, no. 2, pp. 121–128, 1988.

[68] W. Pratt *et al.*, “Digital image processing,” *New York*, pp. 429–32, 1978.

[69] A. Goshtasby and G. Stockman, "Point pattern matching using convex hull edges." *IEEE TRANS. SYST. MAN CYBER.*, vol. 15, no. 5, pp. 631–636, 1985.

[70] G. Stockman, S. Kopstein, and S. Benett, "Matching images to models for registration and object detection via clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 229–241, 1982.

[71] J. Davis and M. Keck, "OSU Registration Algorithm," *Internal Report, Ohio State University, USA*.

[72] O. Mendoza, G. Arnold, and P. Stiller, "Further exploration of the object-image metric with image registration in mind," in *Proceedings of the SPIE, Symposium on Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, B. V. Dasarathy, Ed., vol. 6974, April 2008, pp. 697 405–697 405–12.

[73] Y. Ma, S. Soatto, J. Kosecka, Y. Ma, S. Soatta, J. Kosecka, and S. Sastry, *An invitation to 3-D vision*. Springer, 2004.

[74] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *International joint conference on artificial intelligence*, vol. 3, 1981, p. 3.

[75] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[76] C. Paulson, S. Ezekiel, and D. Wu, "Wavelet-based image registration," in *Proceedings of SPIE Defense & Security Symposium*, vol. 7704, Orlando, FL, USA, April 2010.

[77] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and vision computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[78] L. Brown, "A survey of image registration techniques," *ACM computing surveys (CSUR)*, vol. 24, no. 4, p. 376, 1992.

[79] A. Goshtasby, *2-D and 3-D image registration for medical, remote sensing, and industrial applications*. Wiley-Interscience, 2005.

[80] R. Gonzalez and R. Woods, "Digital Image Processing. ISBN: 9780131687288," 2008.

[81] J. Parker, R. Kenyon, and D. Troxel, "Comparison of interpolating methods for image resampling," *IEEE Trans. Med. Imaging*, vol. 2, no. 1, pp. 31–39, 1983.

[82] E. Stollnitz, T. DeRose, and D. Salesin, “Wavelets for computer graphics: A primer, part 1,” *IEEE Computer Graphics and Applications*, vol. 15, pp. 75–75, 1995.

[83] S. Walker James, “A Primer on WAVELETS and their Scientific Applications,” *USA, Universiti of Wisconsin*, p. 300, 2008.

[84] J. Le Moigne, W. Campbell, and R. Cromp, “An automated parallel image registration technique based on the correlation of wavelet features,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 8, pp. 1849–1864, 2002.

[85] L. Fonseca and M. Costa, “Automatic registration of satellite images,” in *Proceedings of the Brazilian Symposium on Computer Graphic and Image Processing, Brazil*. Citeseer, 1997, pp. 219–226.

[86] Q. Zheng and R. Chellappa, “A computational vision approach to image registration,” *IEEE Transactions on Image Processing*, vol. 2, no. 3, pp. 311–326, 1993.

[87] H. Li and Y. Zhou, “Automatic EO/IR sensor image registration,” in *IEEE Int. Conf. on Image Proc., volume B*, 1995, pp. 161–164.

[88] M. Corvi and G. Nicchiotti, “Multiresolution image registration,” in *Image Processing, 1995. Proceedings., International Conference on*, vol. 3, 1995.

[89] M. Unser, A. Aldroubi, and C. Gerfen, “A multiresolution image registration procedure using spline pyramids,” *Proceedings of the SPIE-Mathematical Imaging: Wavelets and Applications in Signal and Image Processing*, vol. 2034, pp. 160–170, 1993.

[90] J. Djamdji, A. Bijaoui, and R. Maniere, “Geometrical registration of images: the multiresolution approach,” *Photogrammetric Engineering and Remote Sensing*, vol. 59, no. 5, pp. 645–653, 1993.

[91] A. Quddus and O. Basir, “Wavelet-Based Medical Image Registration for Retrieval Applications,” in *Proceedings of the 2008 International Conference on BioMedical Engineering and Informatics-Volume 02*. IEEE Computer Society, 2008, pp. 301–305.

[92] J. Wu and A. Chung, “Multimodal brain image registration based on wavelet transform using SAD and MI,” *Lecture Notes in Computer Science*, pp. 270–277, 2004.

[93] A. Wong and D. Clausi, “Automatic Registration of Inter-band and Inter-sensor Images using Robust Complex Wavelet Feature Representations,” in *Proc. of 5th IAPR Workshop on Pattern Recognition in Remote Sensing (PRRS 2008)*, 2008.

[94] H. Xishan and C. Zhe, "A wavelet-based multisensor image registration algorithm," in *Signal Processing, 2002 6th International Conference on*, vol. 1, 2002.

[95] C. Bejar and D. Megherbi, "A feature-based image registration algorithm using the multi-resolution approach combined with moments of inertia, with application to ladar imaging," in *Proc. SPIE*, vol. 5909, 2005, pp. 506–516.

[96] S. Li, J. Peng, J. Kwok, J. Zhang, and C. Changsha, "Multimodal registration using the discrete wavelet frame transform," in *Proceedings of the 18th International Conference on Pattern Recognition-Volume 03*. Citeseer, 2006, p. 880.

[97] A. Wong and P. Fieguth, "Fast phase-based registration of multimodal image data," *Signal Processing*, vol. 89, no. 5, pp. 724–737, 2009.

[98] N. Kingsbury, "Complex wavelets for shift invariant analysis and filtering of signals," *Applied and Computational Harmonic Analysis*, vol. 10, no. 3, pp. 234–253, 2001.

[99] J. Fauqueur, N. Kingsbury, and R. Anderson, "Multiscale keypoint detection using the dual-tree complex wavelet transform," in *Proceedings of IEEE International Conference on Image Processing*. Citeseer, 2006.

[100] F. Elfouly, M. Mahmoud, and S. Deyab, "Comparison between Haar and Daubechies Wavelet Transformations on FPGA Technology," *International Journal of Computer, Information, and Systems Science, and Engineering*, vol. 2, no. 1, pp. 1047–1061, 2006.

[101] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," 1981.

[102] B. Han, C. Paulson, and D. Wu, "3d surface recovery via deterministic annealing based piecewise linear surface fitting algorithm," Department of Electrical & Computer Engineering, University of Florida, Gainesville, Florida, Tech. Rep., June 2010.

[103] A. Likas, N. Vlassis, *et al.*, "The global k-means clustering algorithm," *Pattern Recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[104] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, 1998.

[105] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: a kernel approach," *Machine Learning*, vol. 74, no. 1, pp. 1–22, 2009.

[106] S. Kirkpatrick, “Optimization by simulated annealing: Quantitative studies,” *Journal of Statistical Physics*, vol. 34, no. 5, pp. 975–986, 1984.

[107] A. Rao, D. Miller, K. Rose, and A. Gersho, “A deterministic annealing approach for parsimonious design of piecewise regression models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 2, pp. 159–173, 1999.

[108] F. Camastra and A. Verri, “A Novel Kernel Method for Clustering,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pp. 801–804, 2005.

[109] B. Han, C. Paulson, and D. Wu, “An automatic surface fitting method for 3d reconstruction from 2d video sequence,” Department of Electrical & Computer Engineering, University of Florida, Gainesville, Florida, Tech. Rep., June 2010.

[110] Z. Zhang, “A flexible new technique for camera calibration,” *Ieee Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[111] C. Strecha, T. Tuytelaars, and L. Van Gool, “Dense matching of multiple wide-baseline views,” in *International Conference on Computer Vision*, vol. 2. Citeseer, 2003, pp. 1194–1201.

[112] M. Lhuillier and L. Quan, “A quasi-dense approach to surface reconstruction from uncalibrated images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 418–433, 2005.

[113] H. Jin, S. Soatto, and A. Yezzi, “Multi-view stereo reconstruction of dense shape and complex appearance,” *International Journal of Computer Vision*, vol. 63, no. 3, p. 189, 2005.

[114] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press New York, NY, USA, 2003.

[115] E. Trucco and A. Verri, *Introductory techniques for 3-D computer vision*. Prentice Hall New Jersey, 1998.

[116] Y. Ma, S. Soatto, and J. Košecká, *An invitation to 3-d vision: from images to geometric models*. Springer Verlag, 2004.

[117] H. Li, B. Adams, L. Guibas, and M. Pauly, “Robust single-view geometry and motion reconstruction,” in *ACM SIGGRAPH Asia 2009 papers*. ACM, 2009, pp. 1–10.

[118] P. Beardsley, P. Torr, and A. Zisserman, “3D Model Acquisition from Extended Image Sequences,” in *Proceedings of the 4th European Conference on Computer Vision-Volume II-Volume II.* Springer-Verlag, 1996, pp. 683–695.

[119] A. Fitzgibbon and A. Zisserman, “Automatic Camera Recovery for Closed or Open Image Sequences,” in *Proceedings of the 5th European Conference on Computer Vision-Volume I-Volume I.* Springer-Verlag, 1998, pp. 311–326.

[120] M. Pollefeys, R. Koch, and V. Gool, “Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters,” *Journal of Computer Vision*, 1998.

[121] F. Devernay and O. Faugeras, “Automatic calibration and removal of distortion from scenes of structured environments,” *Investigative and Trial Image Processing*, vol. 2567, pp. 62–72, 1995.

[122] J. Yagnik and K. Ramakrishnan, “A model based factorization approach for dense 3D recovery from monocular video,” in *Seventh IEEE International Symposium on Multimedia*, 2005, p. 4.

[123] V. Popescu, E. Sacks, and G. Bahmutov, “Interactive point-based modeling from dense color and sparse depth,” in *Eurographics Symposium on Point-Based Graphics.* Citeseer, 2004.

[124] H. Chang, J. Moura, Y. Wu, K. Sato, and C. Ho, “Reconstruction of 3D dense cardiac motion from tagged MR sequences,” in *IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2004*, 2004, pp. 880–883.

[125] O. Faugeras and R. Keriven, “Complete Dense Stereovision Using Level Set Methods,” in *Proceedings of the 5th European Conference on Computer Vision-Volume I-Volume I.* Springer-Verlag, 1998, p. 393.

[126] R. Koch, M. Pollefeys, and L. Gool, “Multi Viewpoint Stereo from Uncalibrated Video Sequences,” in *Proceedings of the 5th European Conference on Computer Vision-Volume I-Volume I.* Springer-Verlag, 1998, p. 71.

[127] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool, “Automated reconstruction of 3D scenes from sequences of images,” *ISPRS Journal Of Photogrammetry And Remote Sensing*, vol. 55, no. 4, pp. 251–267, 2000.

[128] M. Lhuillier and L. Quan, “Surface reconstruction by integrating 3D and 2D data of multiple views,” in *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, 2003, pp. 1313–1320.

[129] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *International joint conference on artificial intelligence*, vol. 3. Citeseer, 1981, p. 3.

[130] J. Barron, D. Fleet, and S. Beauchemin, “Performance of optical flow techniques,” *International journal of computer vision*, vol. 12, no. 1, pp. 43–77, 1994.

[131] J. Shi and C. Tomasi, “Good Features to Track,” in *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR’94)*, 1994, pp. 593–600.

[132] B. Han, C. Paulson, and D. Wu, “Depth based image registration via geometric segmentation,” Department of Electrical & Computer Engineering, University of Florida, Gainesville, Florida, Tech. Rep., June 2010.